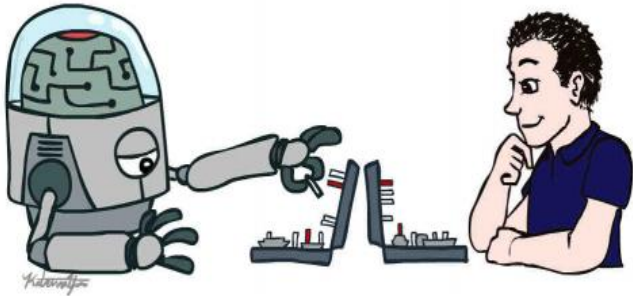
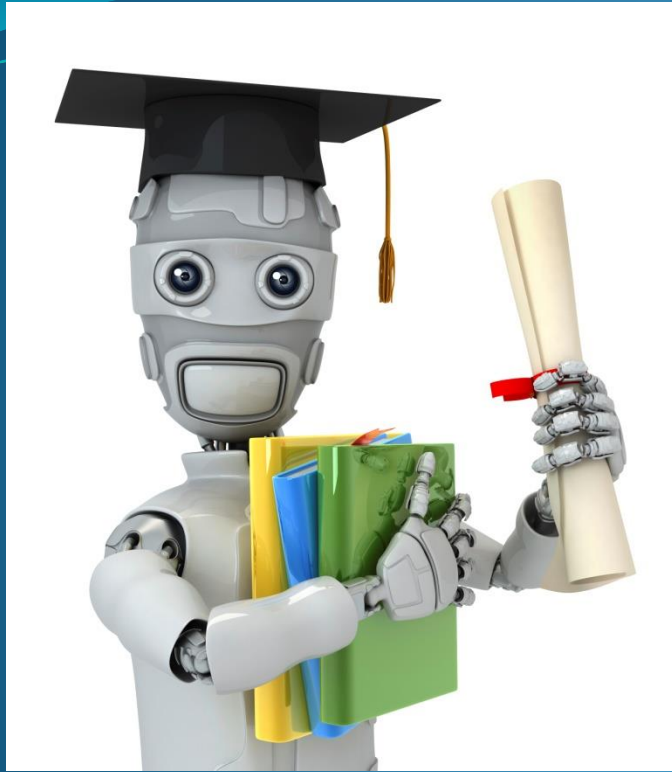


Artificial Intelligence

Part I: Machine learning





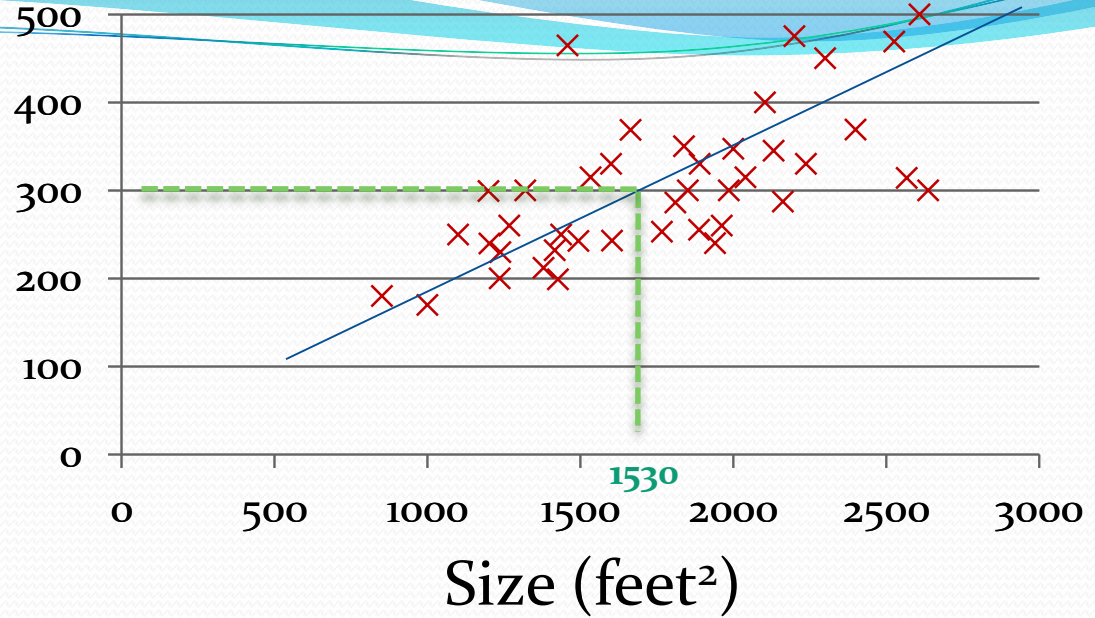
Machine Learning

Linear regression with one variable

Model representation

Housing Prices

Price
(in 1000s
of dollars)



Supervised Learning

Given the “right answer” for each example in the data.

Regression Problem

Predict real-valued output

	Size in feet ² (x)	Price (\$) in 1000's (y)
Training set of housing prices	2104	460
	1416	232
	1534	315
	852	178

Notation:

m = Number of training examples

x's = “input” variable / features

y's = “output” variable / “target” variable

(x, y) is one training example

($x^{(i)}$, $y^{(i)}$) is the i^{th} training example

Training Set



Learning Algorithm



Size of house



h

Estimated price
(Estimated value of y)



x

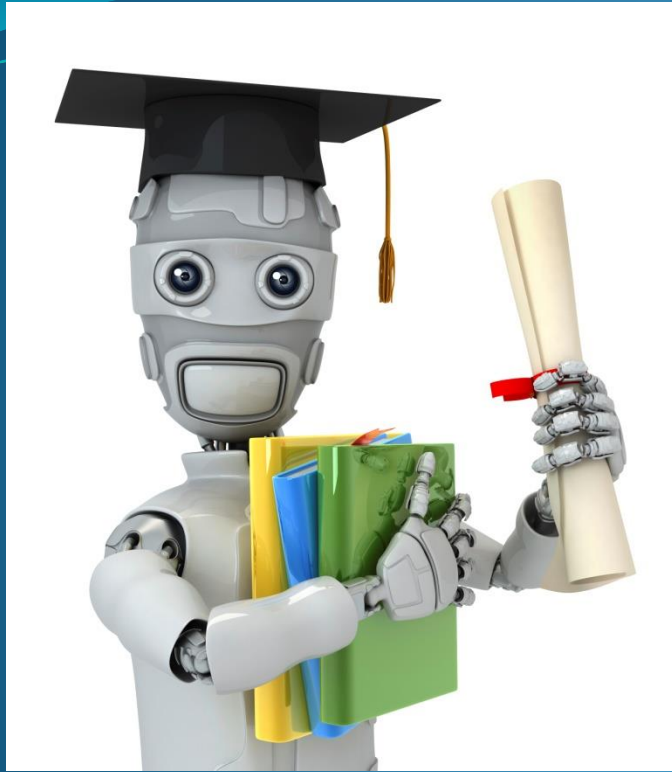
h is a function that maps from x 's to y 's

How do we represent h ?

Hypothesis:

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

Linear regression with one variable.
Univariate linear regression.



Machine Learning

Linear regression with one variable

Cost function

Training Set

Size in feet² (x)

Price (\$) in 1000's (y)

2104

460

1416

232

1534

315

852

178

...

...

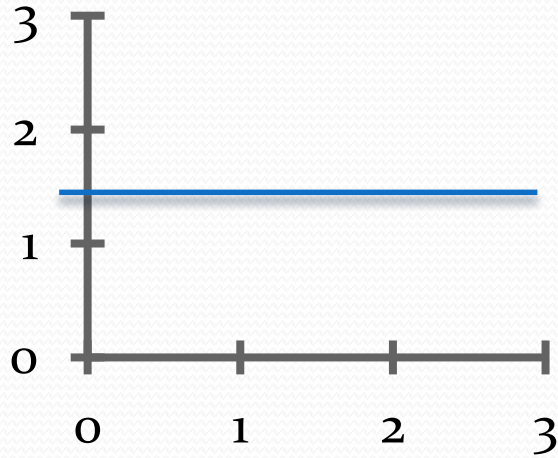
Hypothesis: which we use to make predictions

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

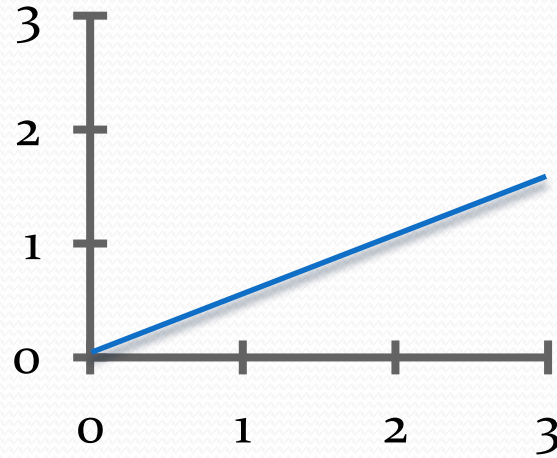
θ_i 's: Parameters

How to choose θ_i 's ? This will let us figure out how to fit the best possible straight line to our data

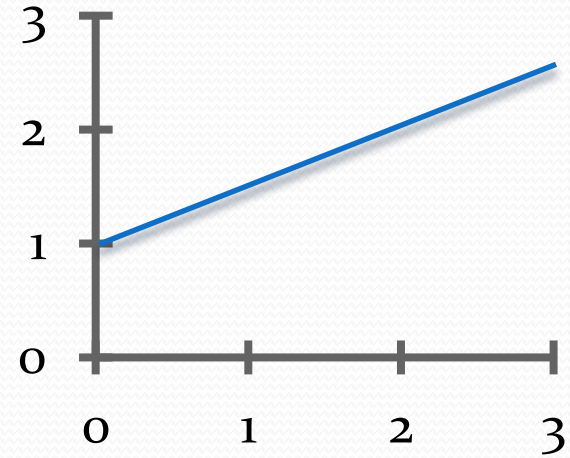
$$h_{\theta}(x) = \theta_0 + \theta_1 x$$



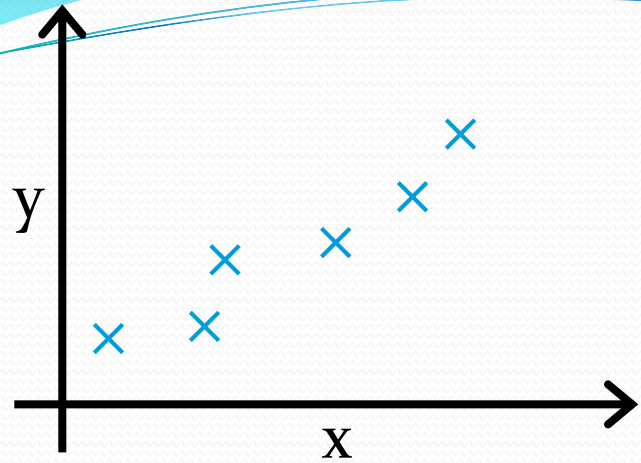
$$\theta_0 = 1.5$$
$$\theta_1 = 0$$



$$\theta_0 = 0$$
$$\theta_1 = 0.5$$



$$\theta_0 = 1$$
$$\theta_1 = 0.5$$



Cost Function:

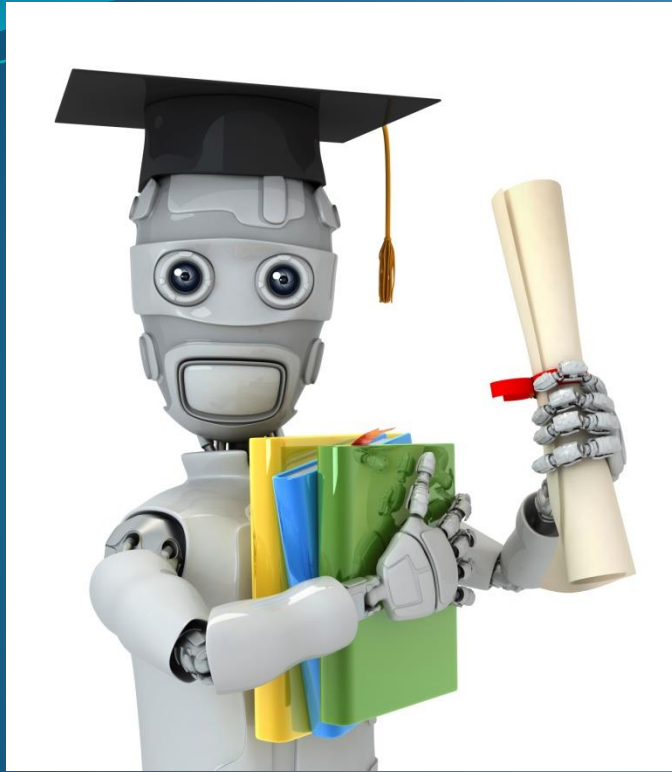
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

Goal: minimize $J(\theta_0, \theta_1)$
 θ_0, θ_1

Idea: Choose θ_0, θ_1 so that $h_{\theta}(x)$ is close to y for our training examples (x, y)



Minimization problem



Machine Learning

Linear regression with one variable

Cost function intuition I

Hypothesis:

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

Parameters:

$$\theta_0, \theta_1$$

Cost Function:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

Goal: minimize $J(\theta_0, \theta_1)$
 θ_0, θ_1

$$h_{\theta}(x) = \theta_1 x$$

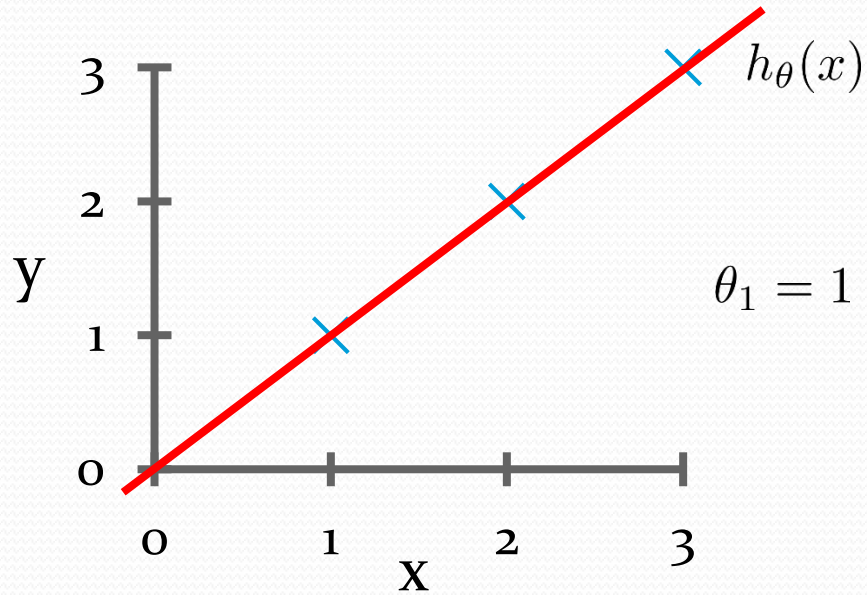
$$\theta_1$$

$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

minimize $J(\theta_1)$
 θ_1

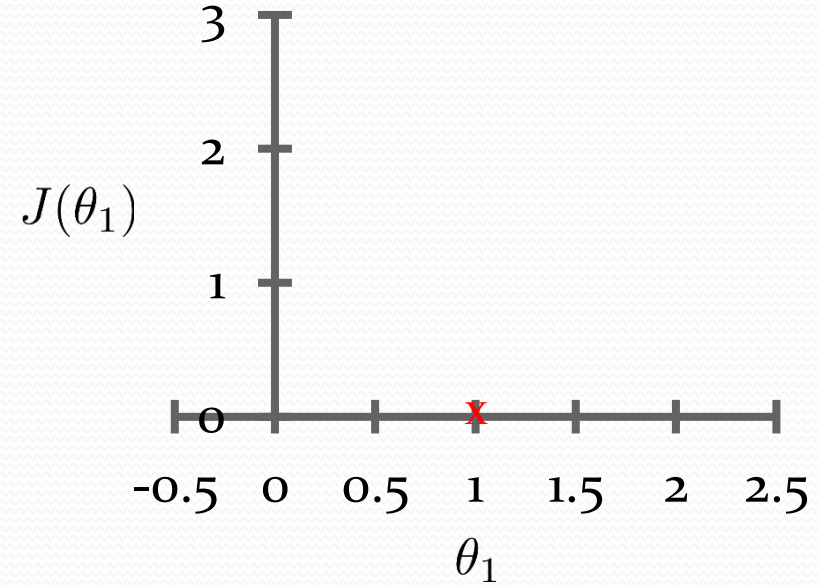
$$h_{\theta}(x)$$

(for fixed θ_1 , this is a function of x)



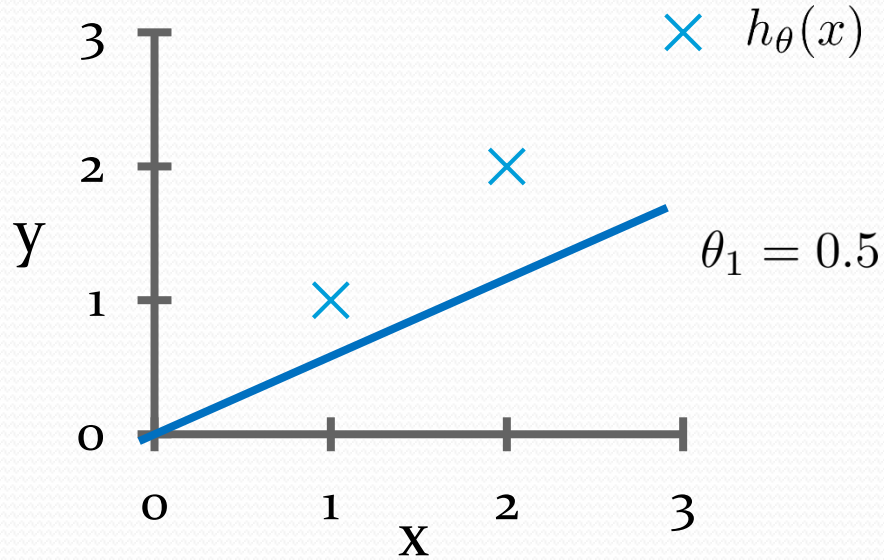
$$J(\theta_1)$$

(function of the parameter θ_1)



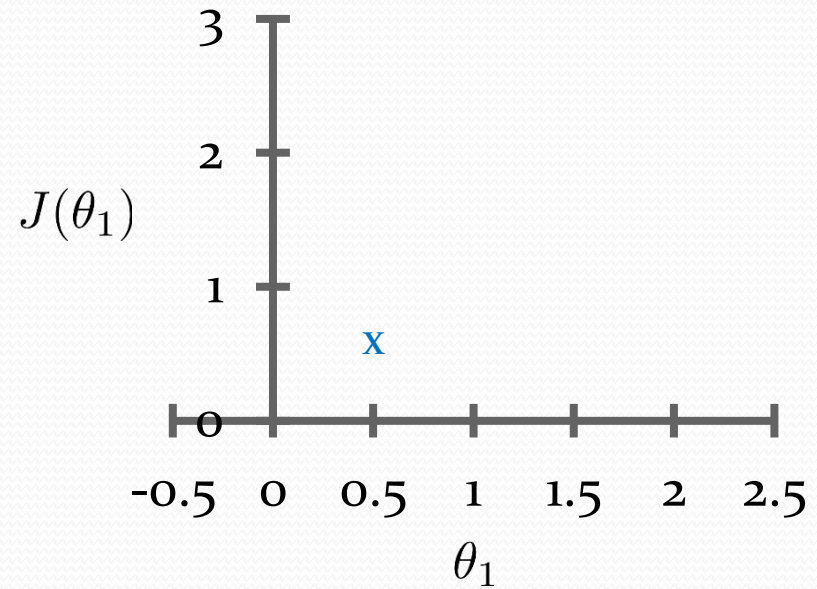
$$h_{\theta}(x)$$

(for fixed θ_1 , this is a function of x)



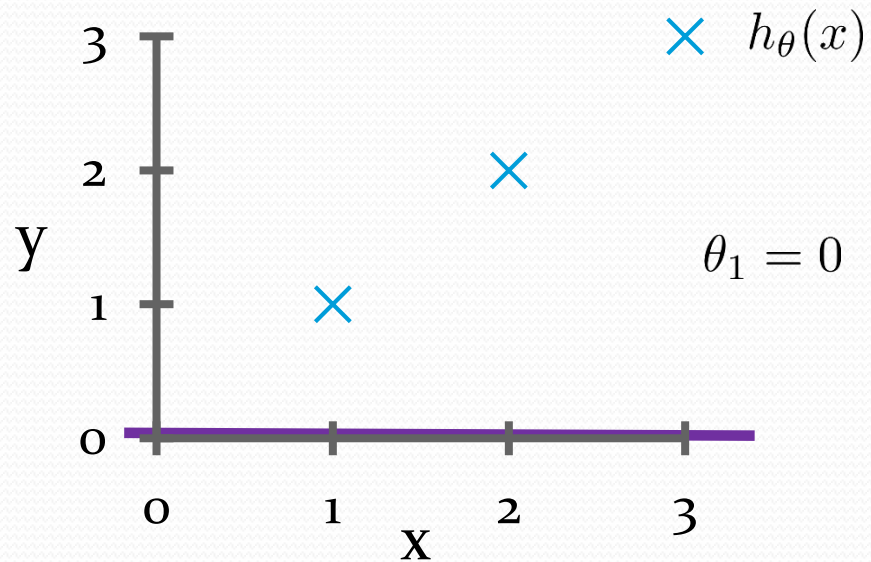
$$J(\theta_1)$$

(function of the parameter θ_1)



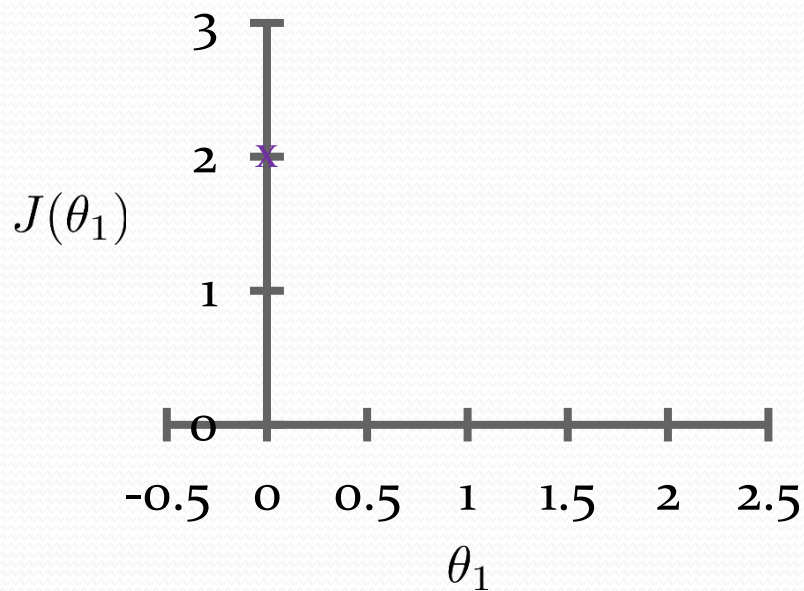
$$h_{\theta}(x)$$

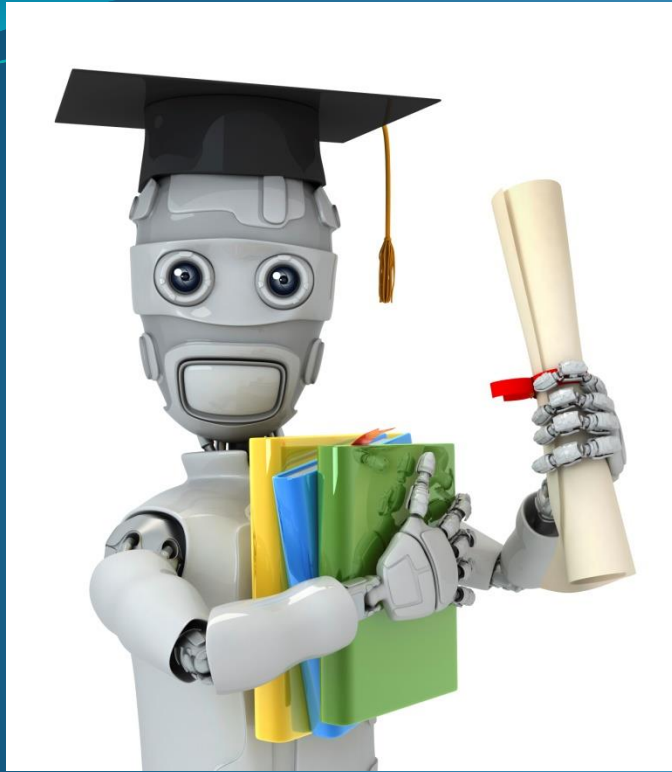
(for fixed θ_1 , this is a function of x)



$$J(\theta_1)$$

(function of the parameter θ_1)





Machine Learning

Linear regression with one variable

Cost function intuition II

Hypothesis: $h_{\theta}(x) = \theta_0 + \theta_1 x$

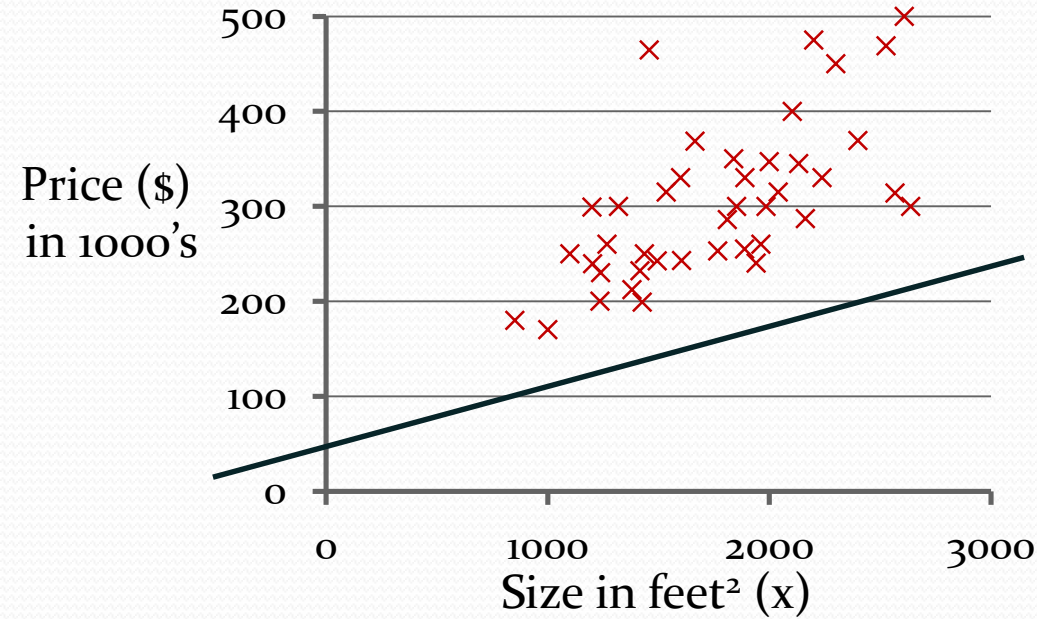
Parameters: θ_0, θ_1

Cost Function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}} J(\theta_0, \theta_1)$

$$h_{\theta}(x)$$

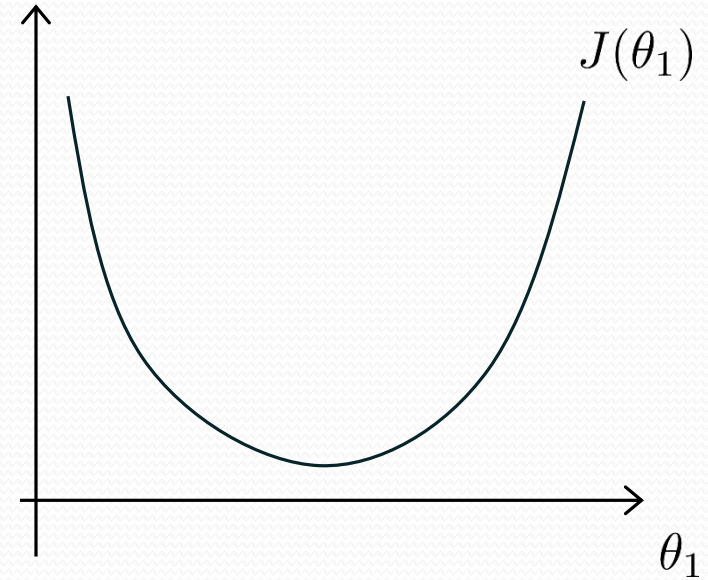
(for fixed θ_0, θ_1 , this is a function of x)

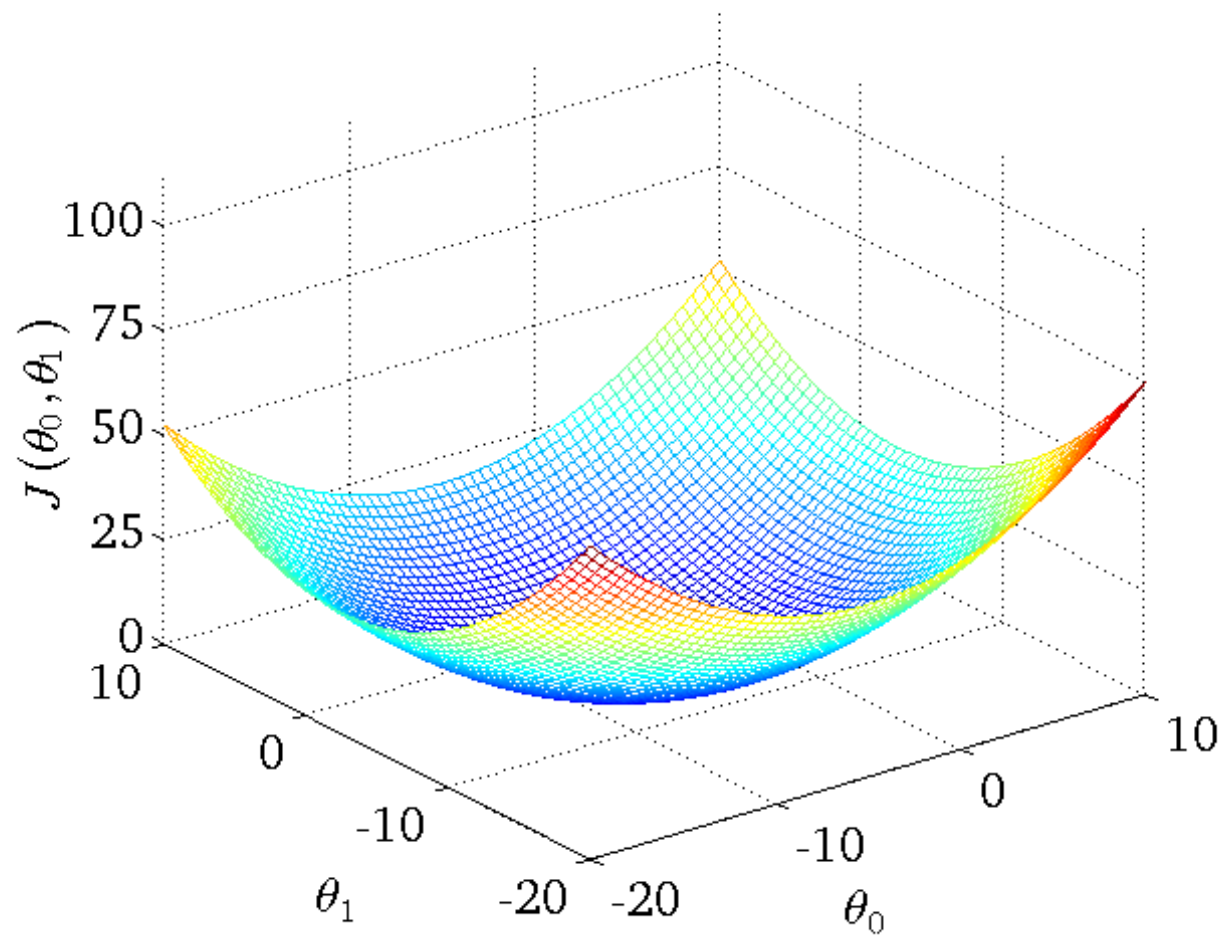


$$h_{\theta}(x) = 50 + 0.06x$$

$$J(\theta_0, \theta_1)$$

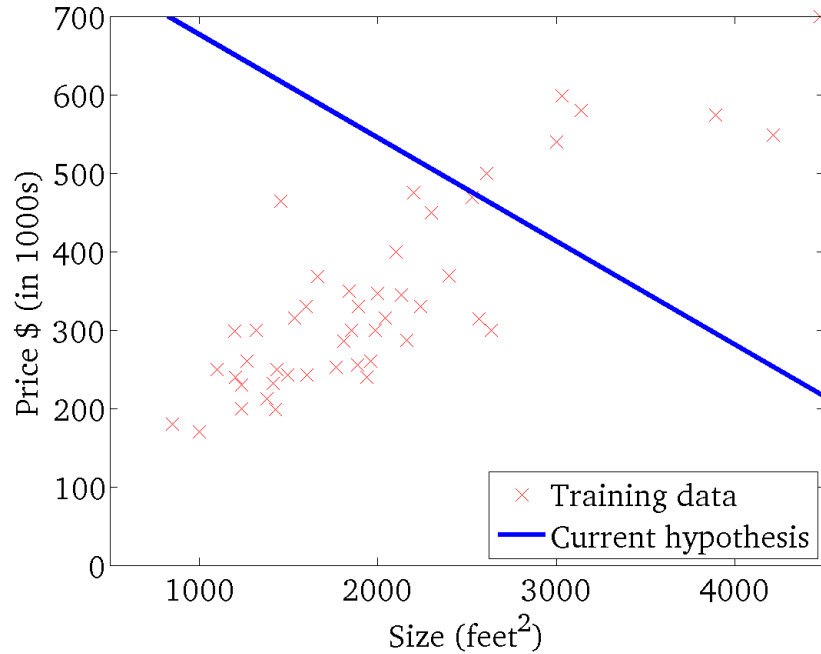
(function of the parameters θ_0, θ_1)





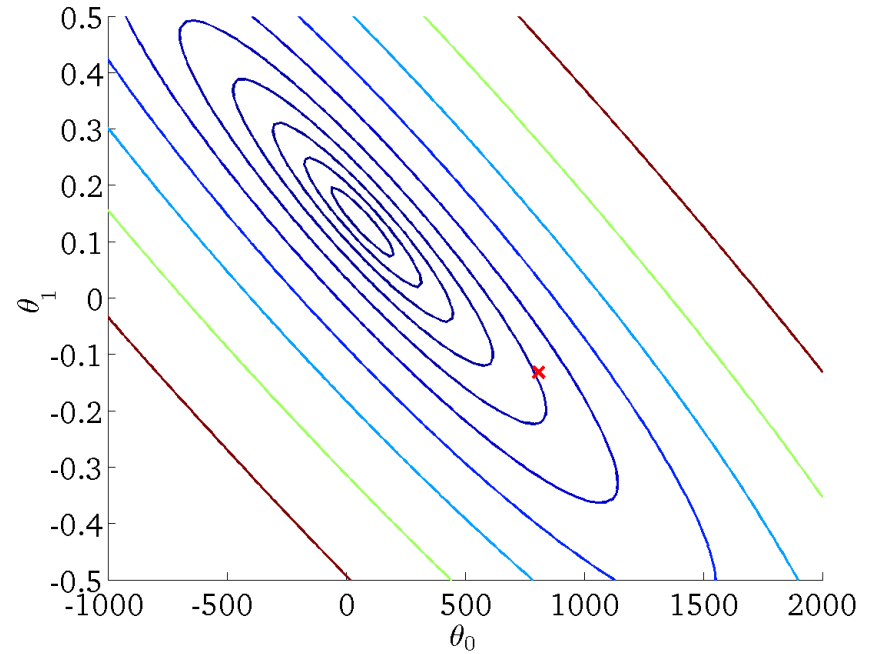
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



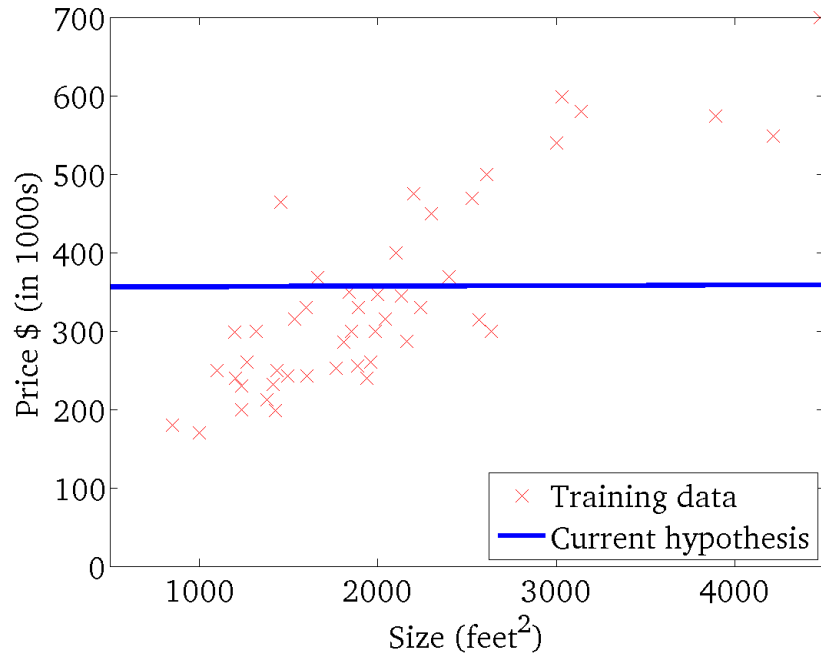
$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



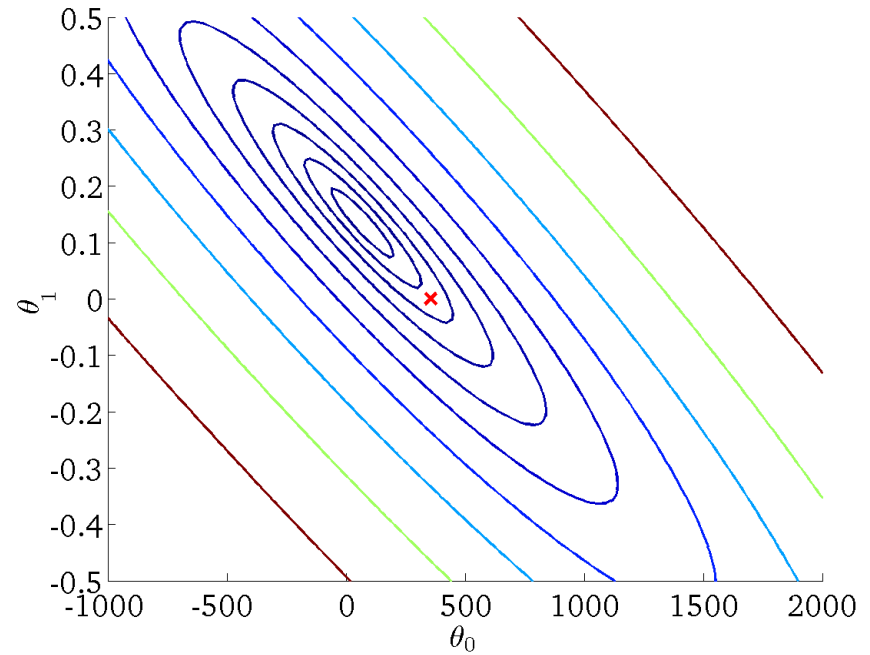
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



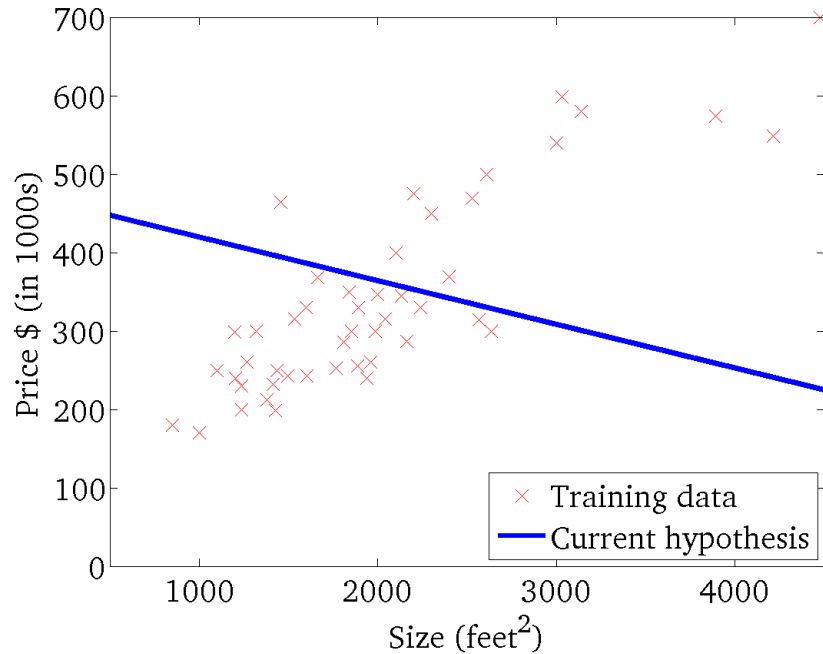
$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



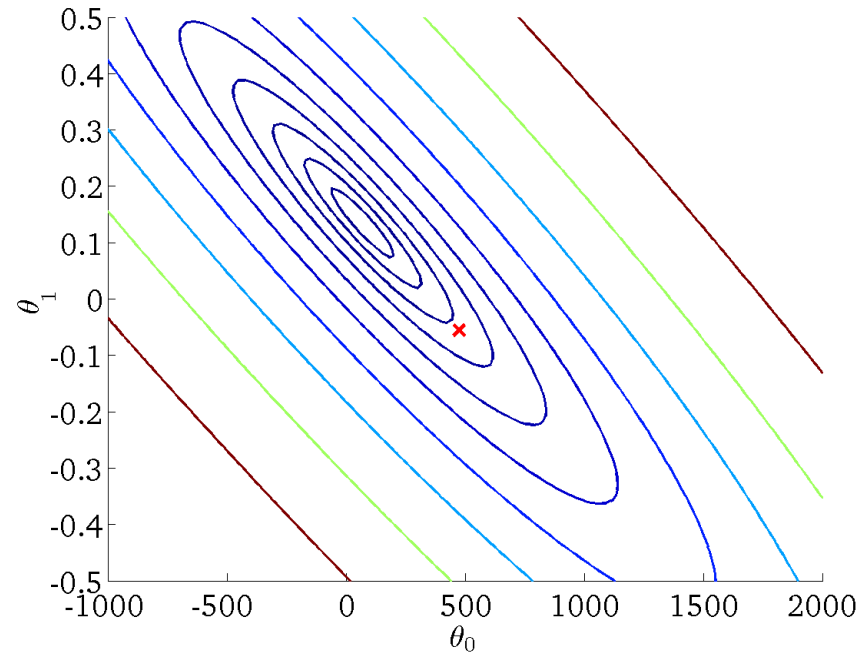
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



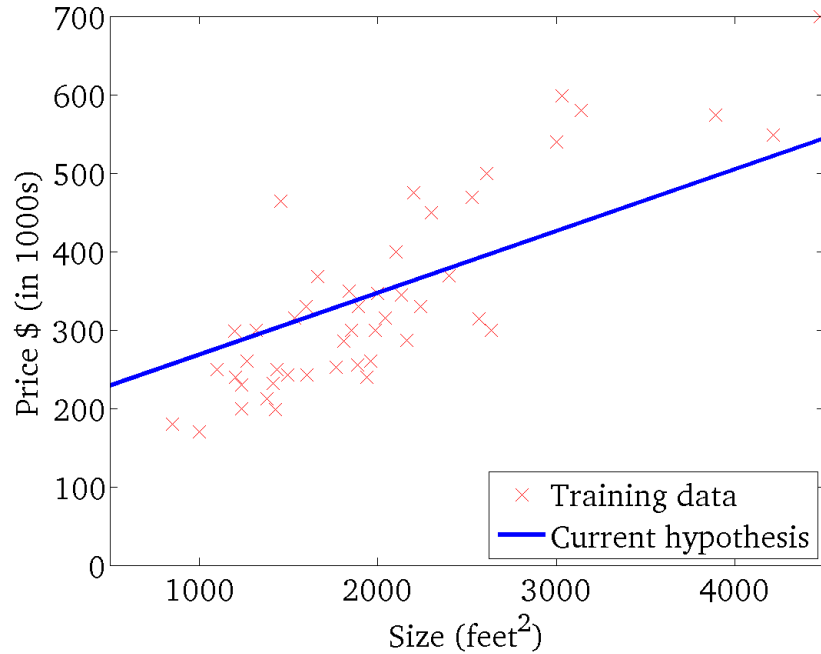
$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



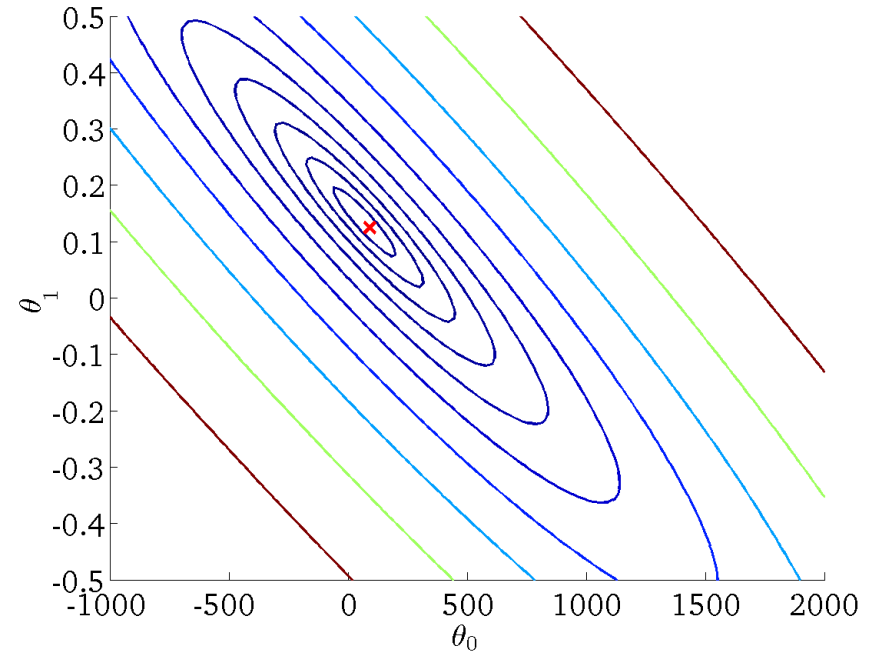
$$h_{\theta}(x)$$

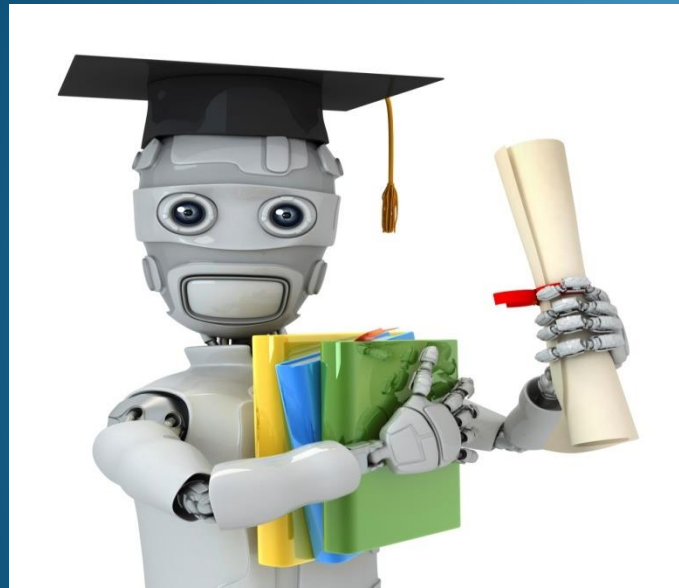
(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



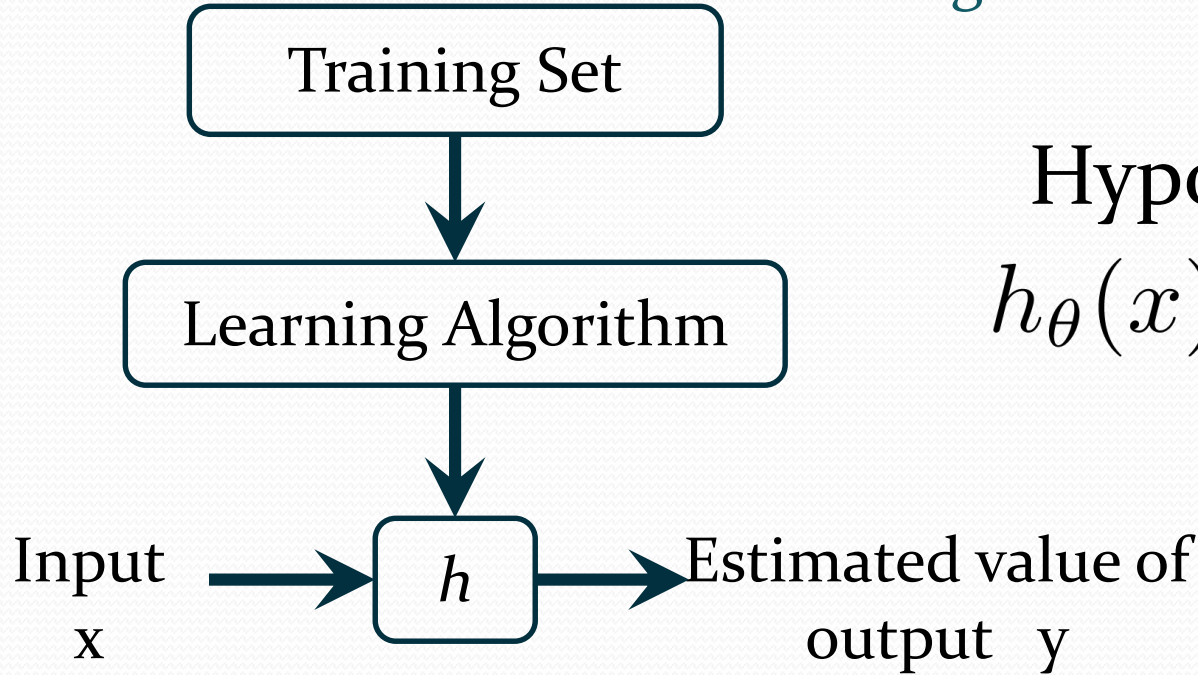


Machine Learning

Linear regression with one variable

Gradient descent

Linear regression with one variable.
Univariate linear regression.



Hypothesis:

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

Hypothesis: $h_{\theta}(x) = \theta_0 + \theta_1 x$

Parameters: θ_0, θ_1

Cost Function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$

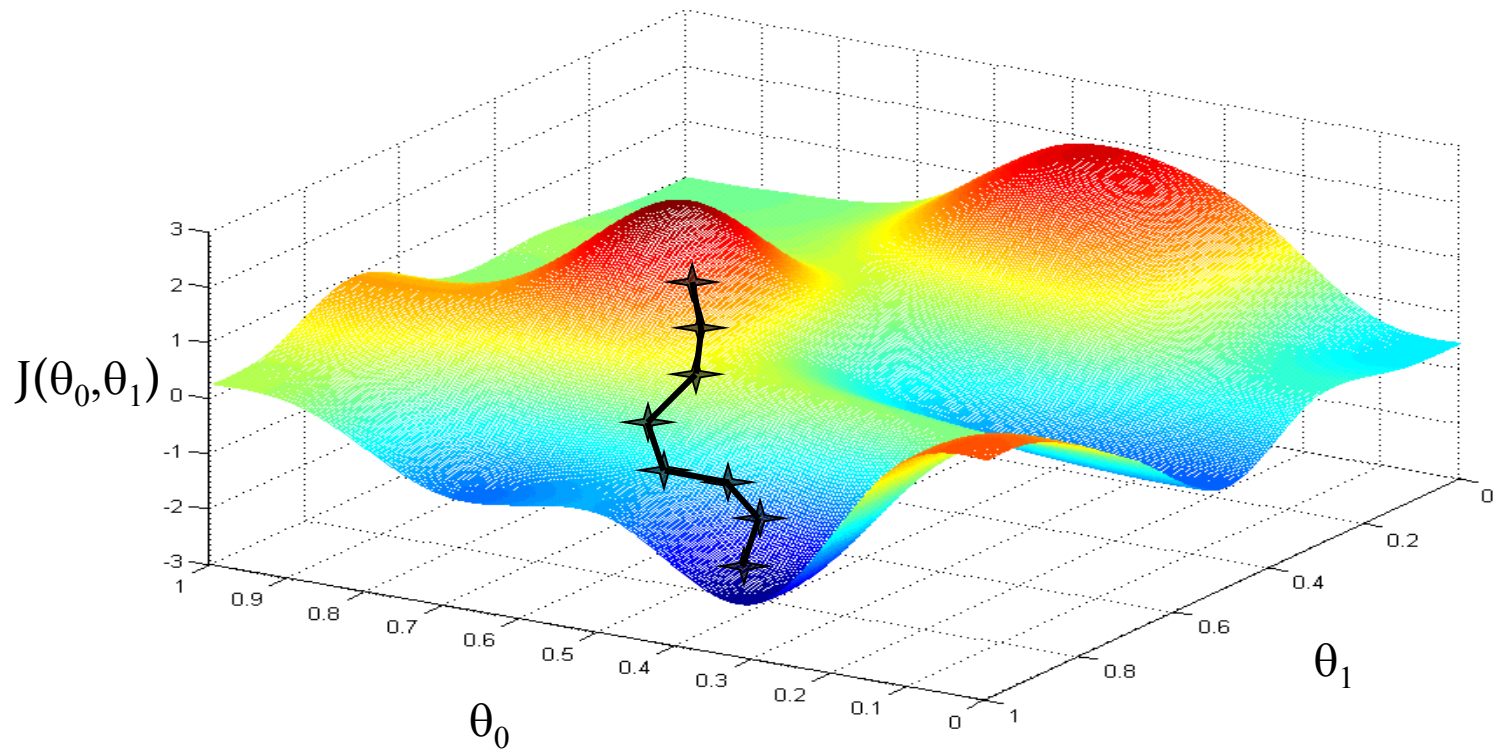
Goal: $\underset{\theta_0, \theta_1}{\text{minimize}} J(\theta_0, \theta_1)$

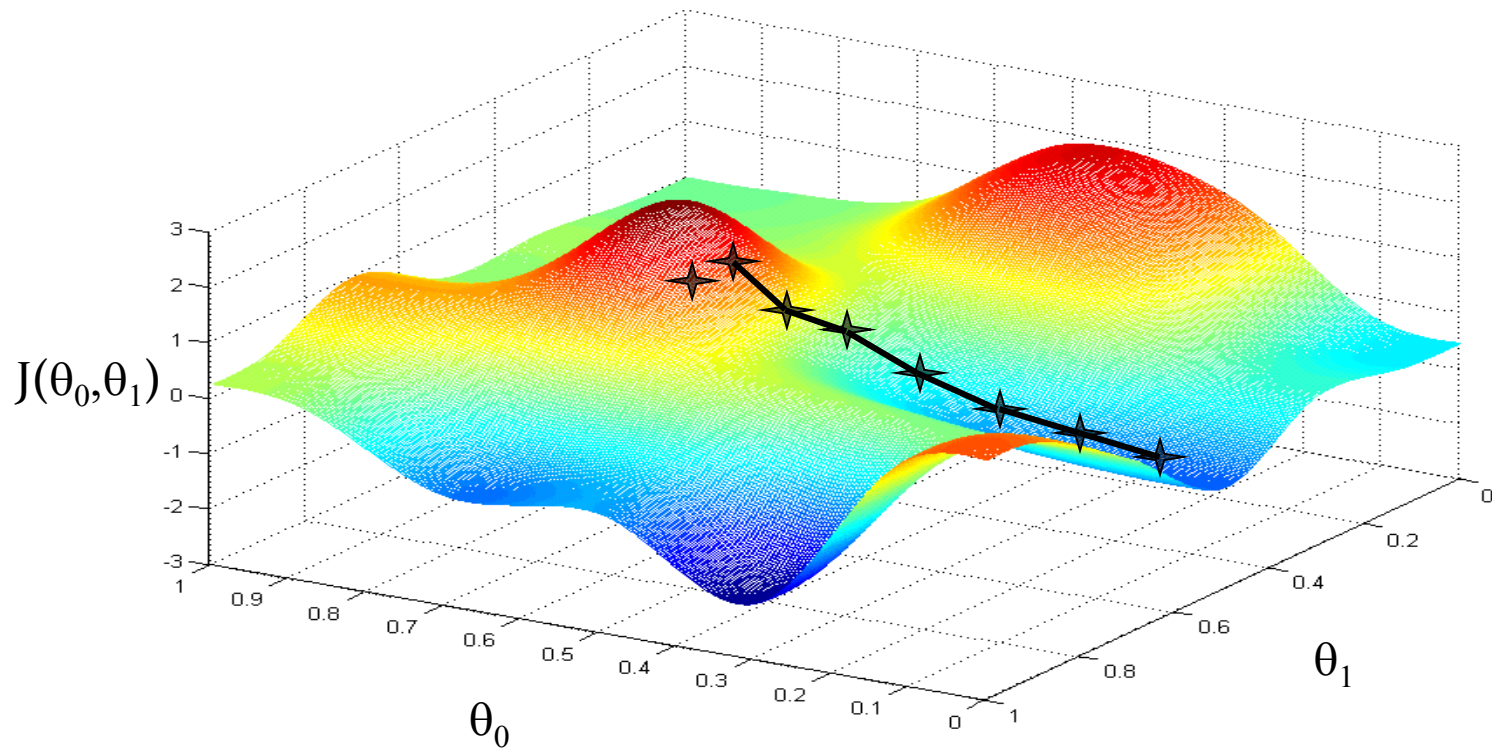
The gradient decent algorithm is used to minimize the objective function

$$\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$$

Outline:

- Start with some initials θ_0, θ_1 $J(\theta_0, \theta_1)$
- Keep changing θ_0, θ_1 to reduce
until we hopefully end up at a minimum





Gradient descent algorithm

means until end up at a local minimum

repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad (\text{for } j = 0 \text{ and } j = 1)$$

}

Learning rate that controls how big step we take down with gradient decent

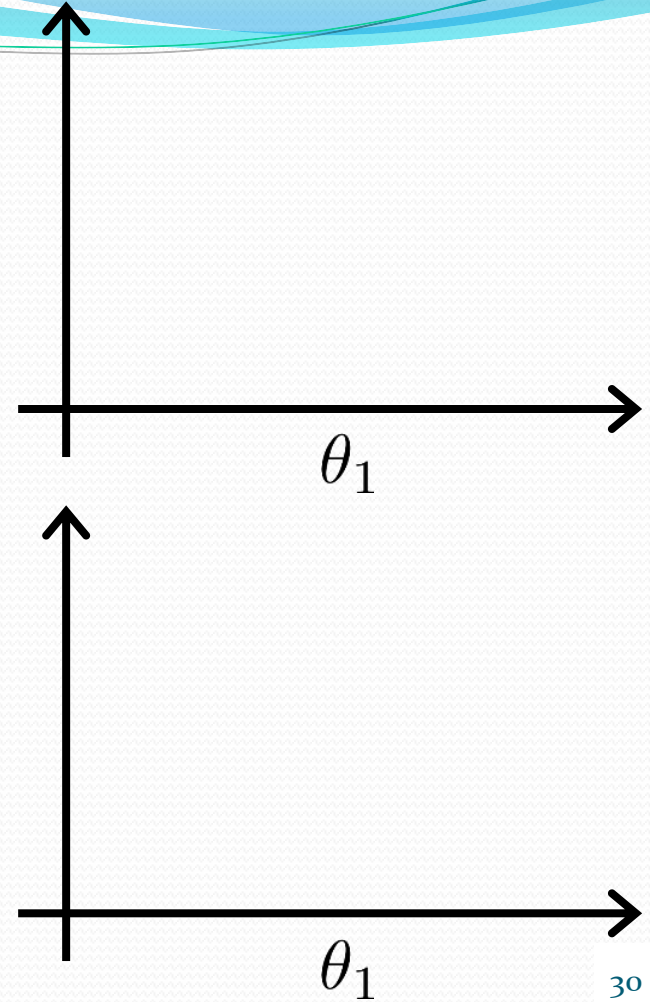
Correct: Simultaneously update

$$\begin{aligned} \text{temp0} &:= \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) \\ \text{temp1} &:= \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) \\ \theta_0 &:= \text{temp0} \\ \theta_1 &:= \text{temp1} \end{aligned}$$

Incorrect:

$$\begin{aligned} \text{temp0} &:= \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) \\ \theta_0 &:= \text{temp0} \\ \text{temp1} &:= \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) \\ \theta_1 &:= \text{temp1} \end{aligned}$$

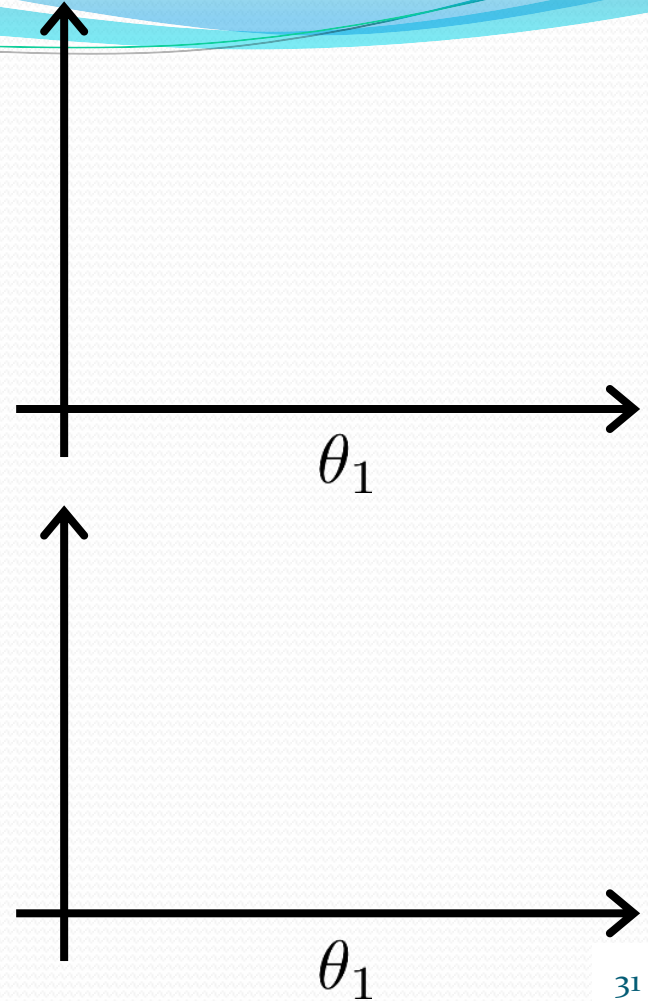
Intuition of derivative term

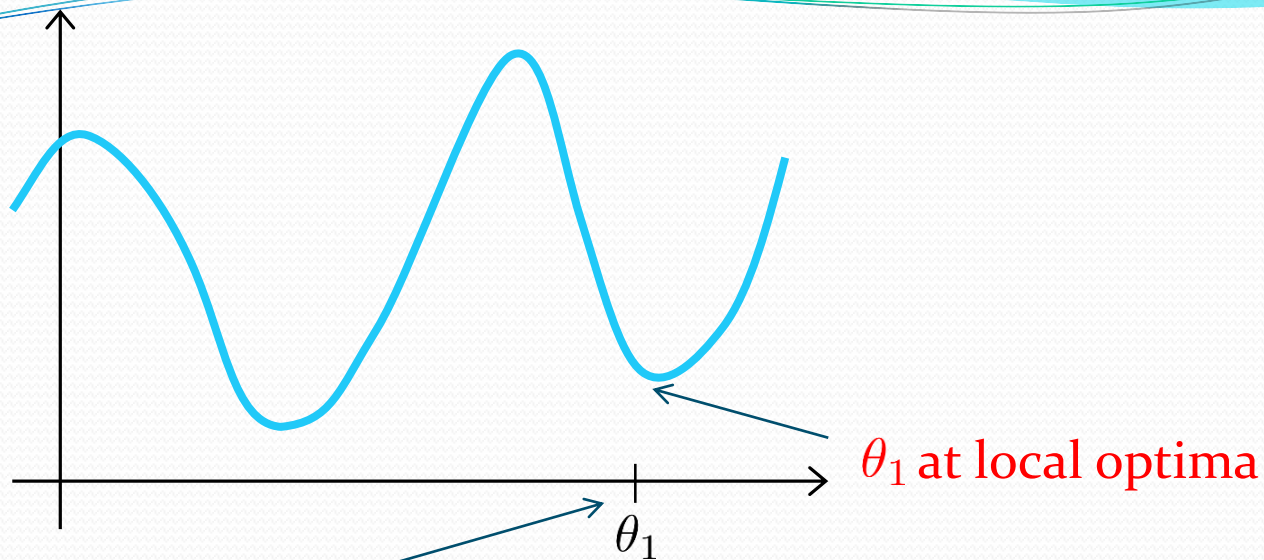


$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

If α is too small, gradient descent can be slow.

If α is too large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge.





Current value of θ_1

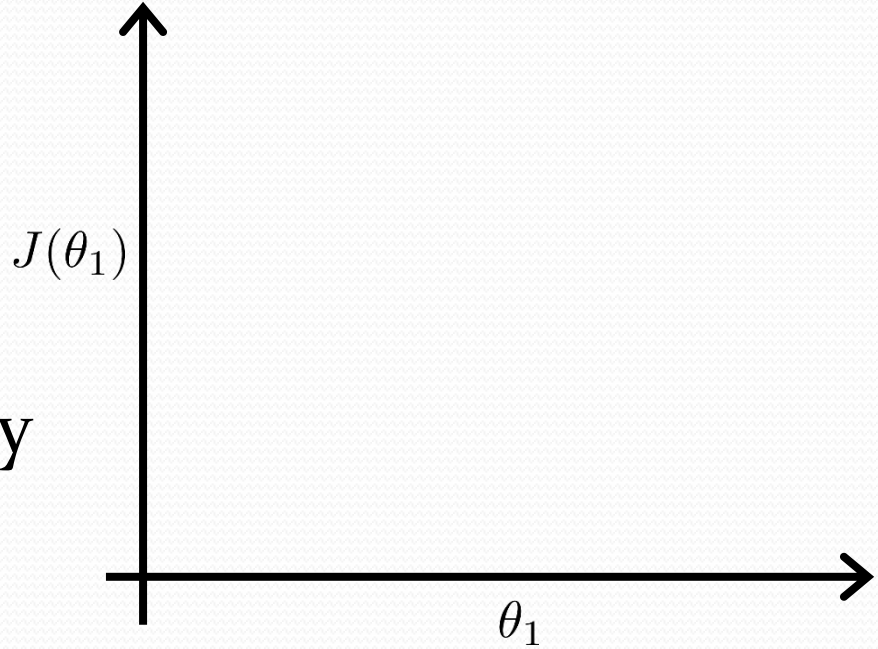
θ_1 at local optima

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$

Gradient descent can converge to a local minimum, even with the learning rate α fixed.

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$

As we approach a local minimum, gradient descent will automatically take smaller steps. So, no need to decrease α over time.



Gradient descent algorithm

repeat until convergence {
 $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$
 (for $j = 1$ and $j = 0$)
}

Linear Regression Model

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

The gradient decent algorithm is used to minimize the objective function

$$\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$$

Gradient descent algorithm

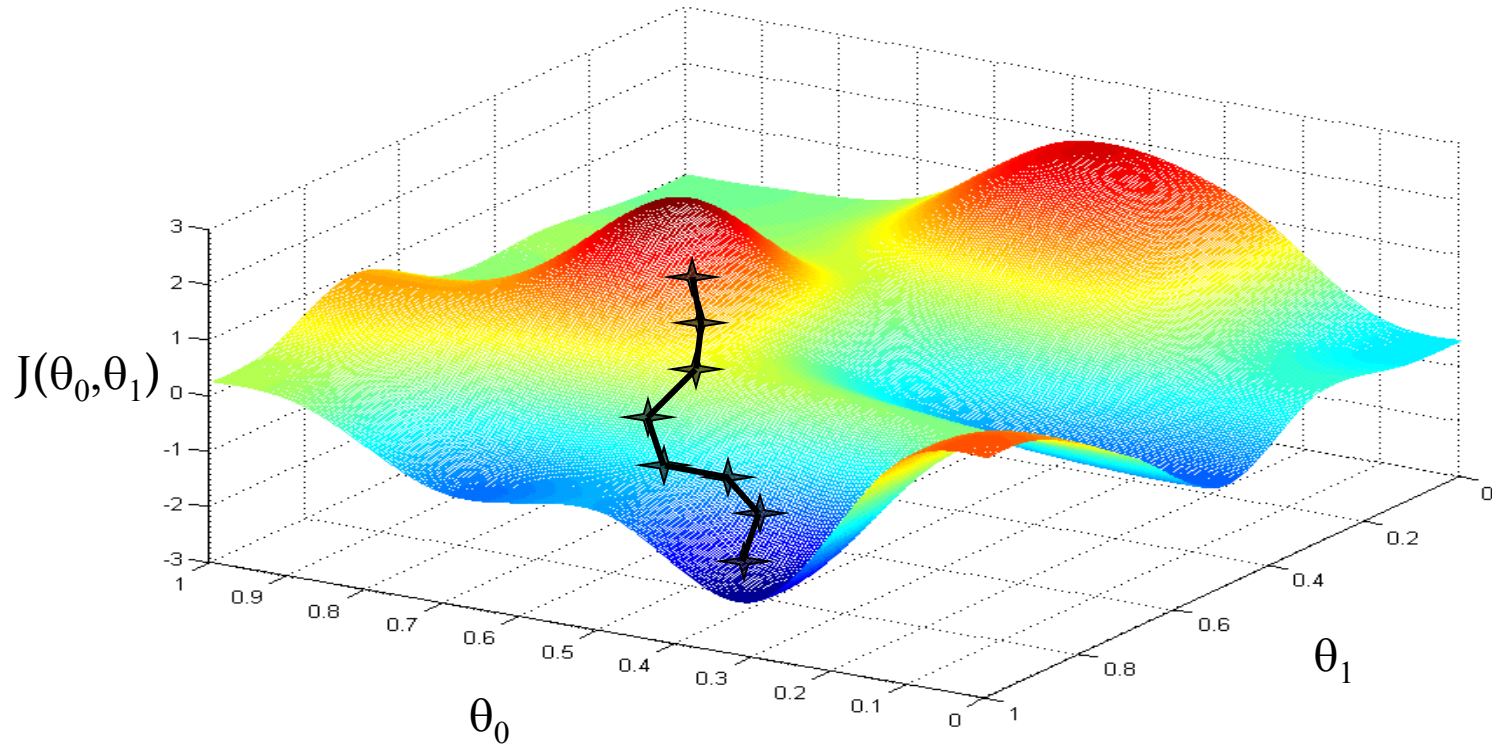
repeat until convergence {

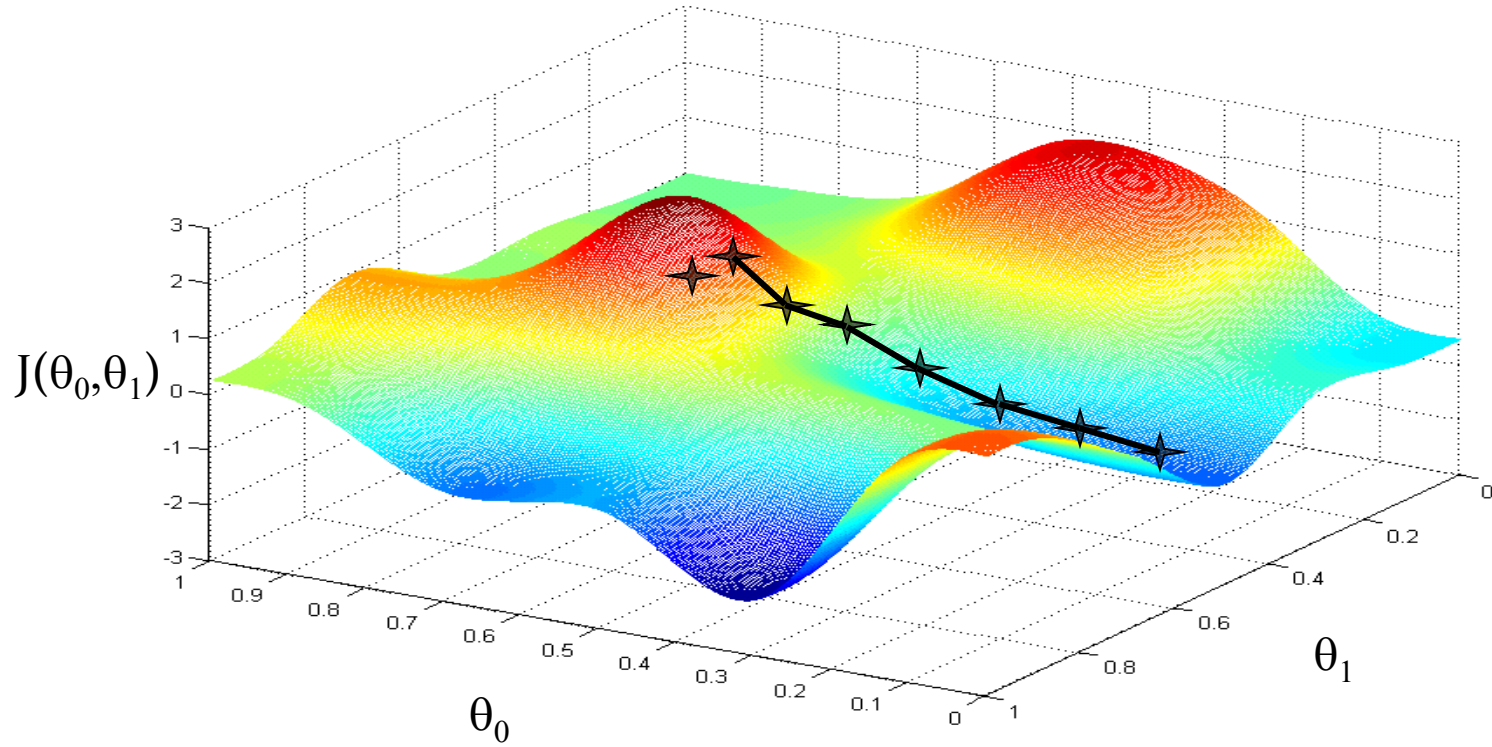
$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) \cdot x^{(i)}$$

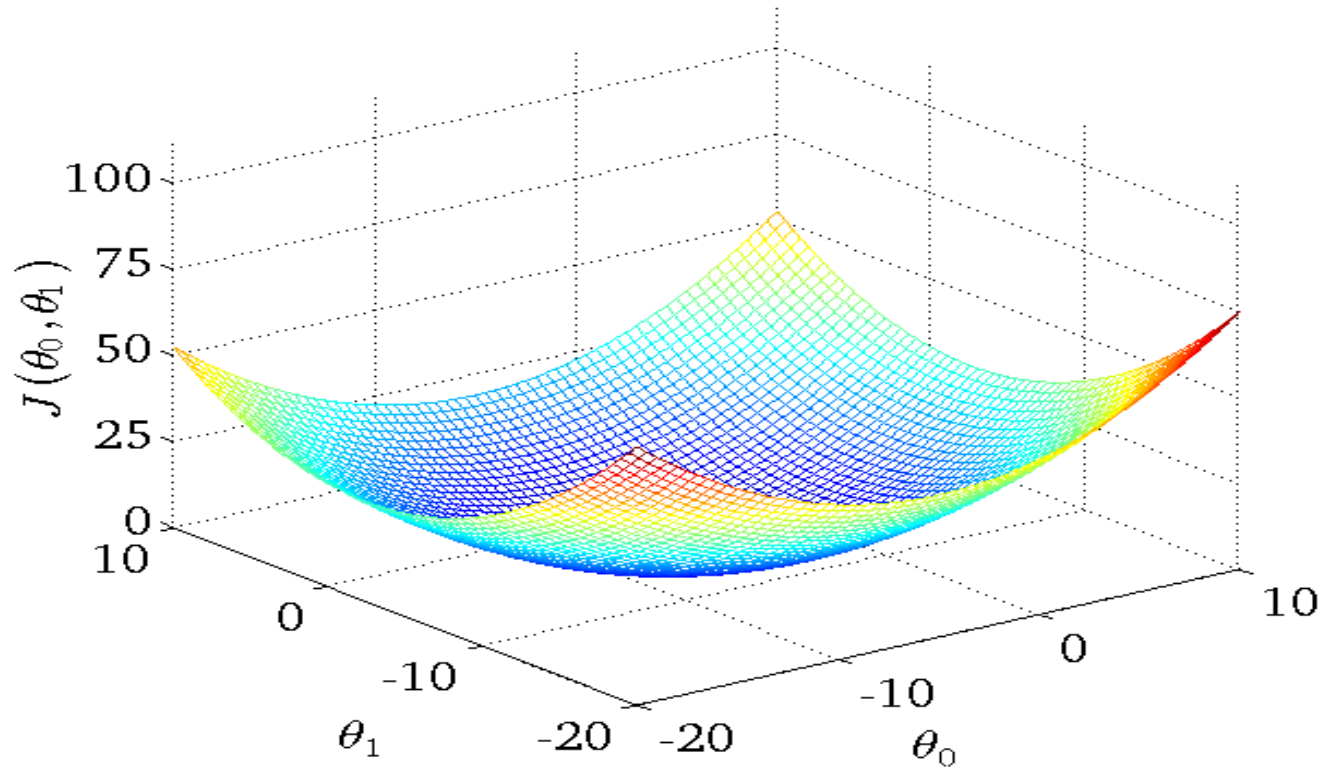
}

} update
 θ_0 and θ_1
simultaneously



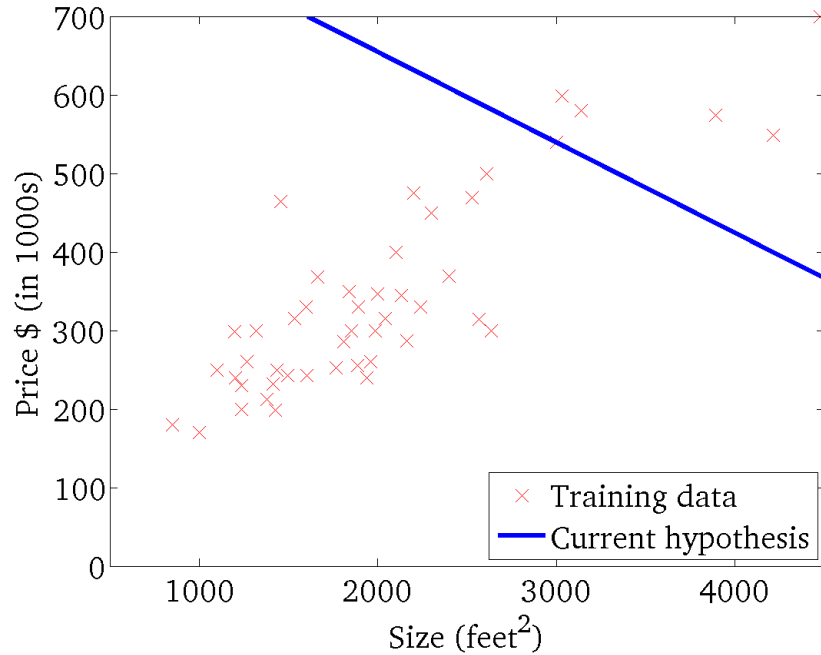


Cost function is always a convex function
Function with one global optima



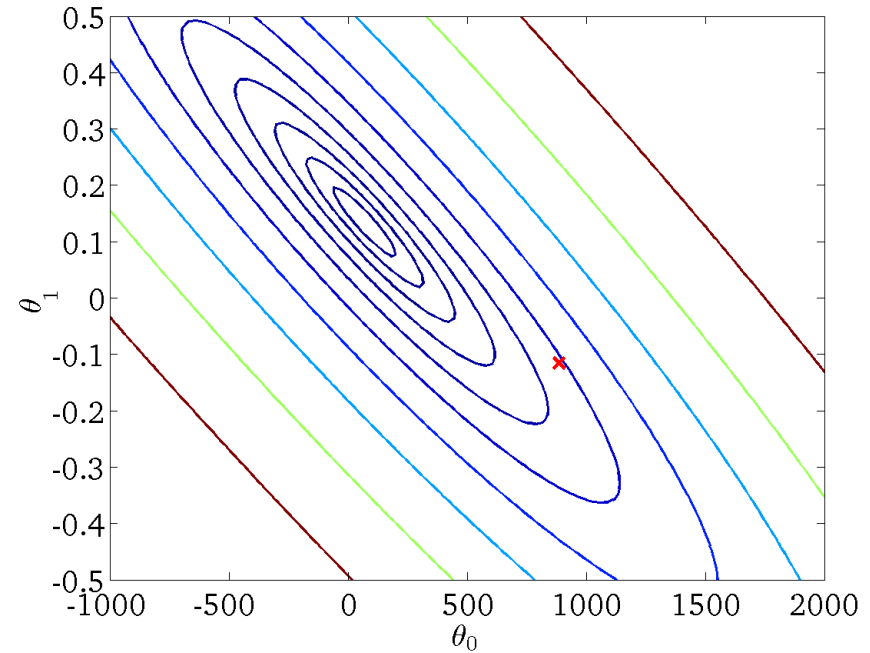
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



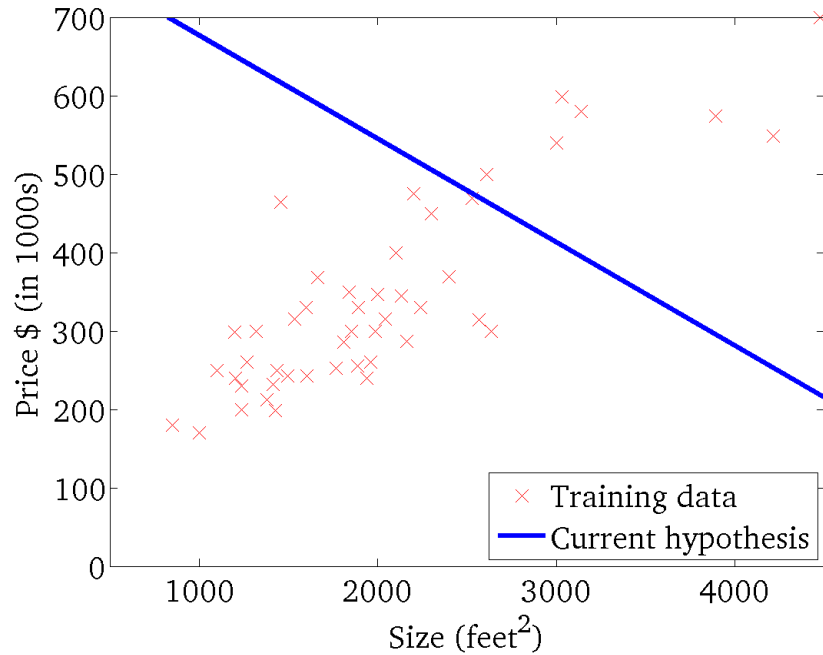
$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



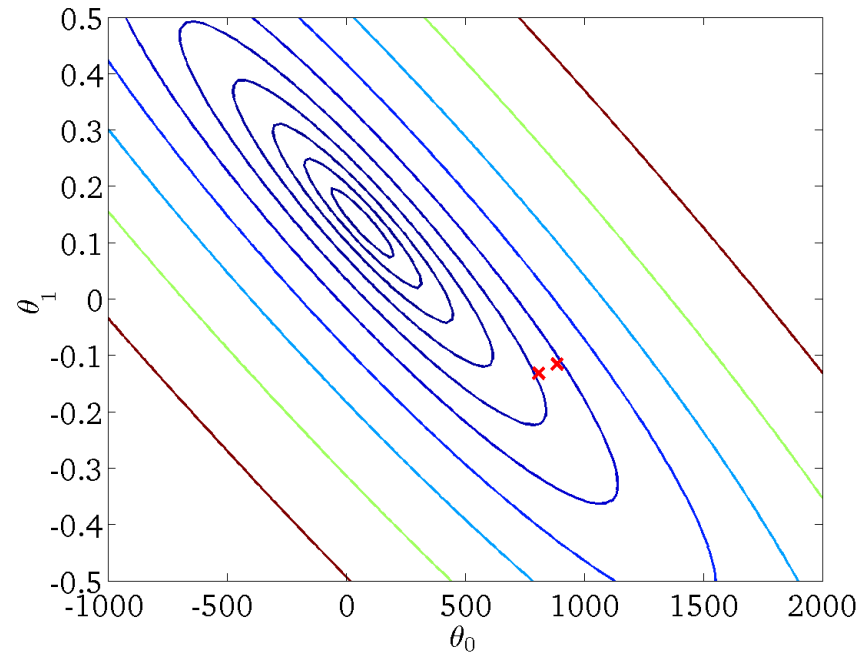
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



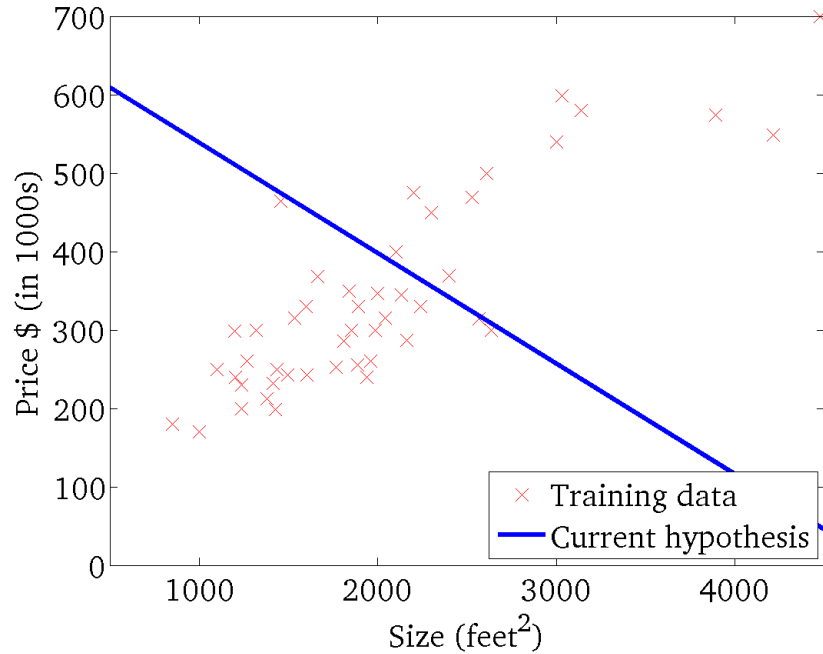
$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



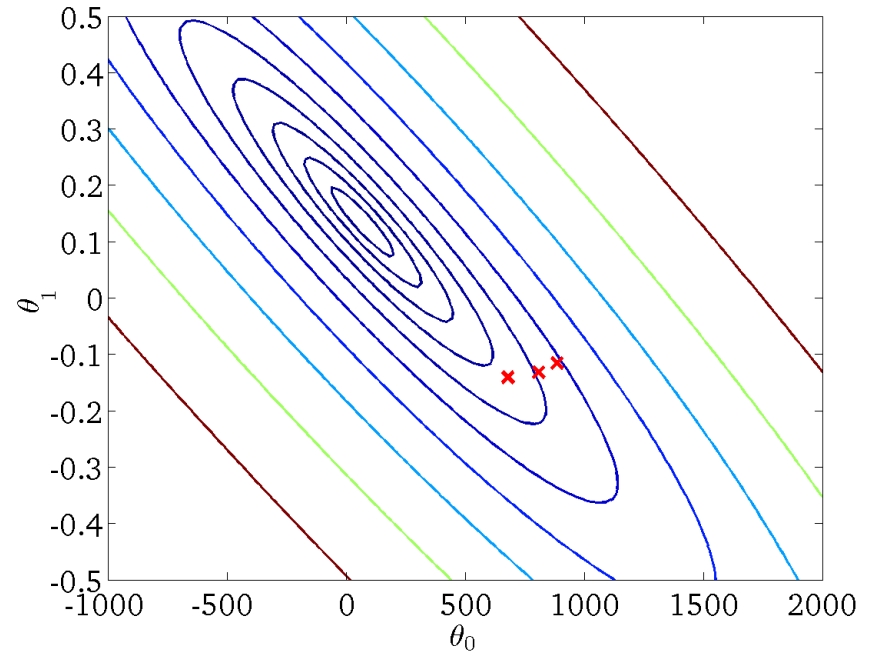
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



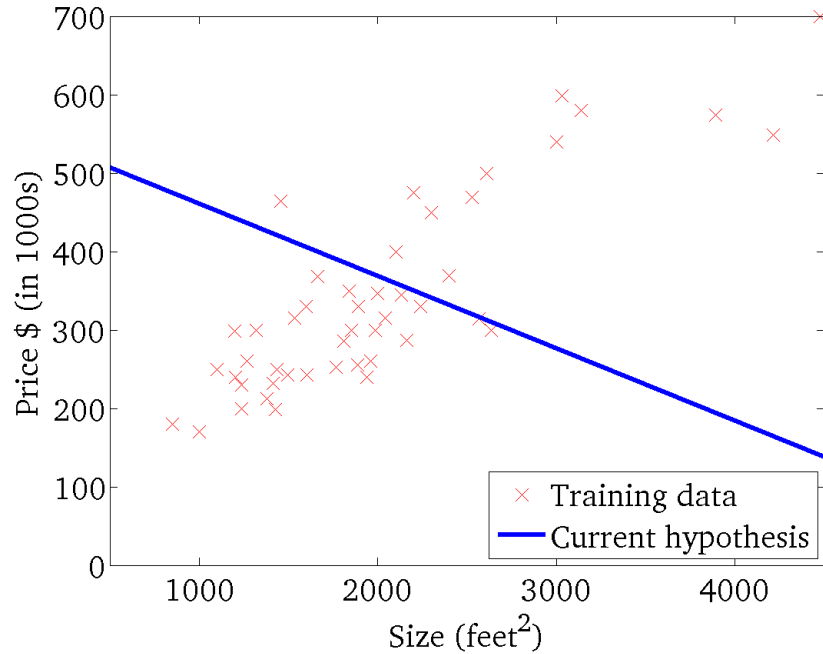
$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



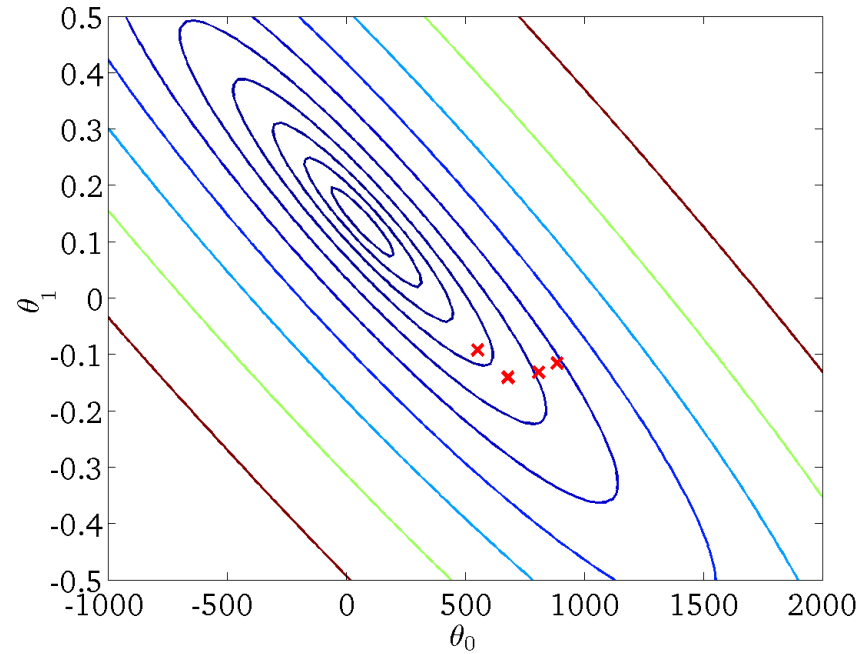
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



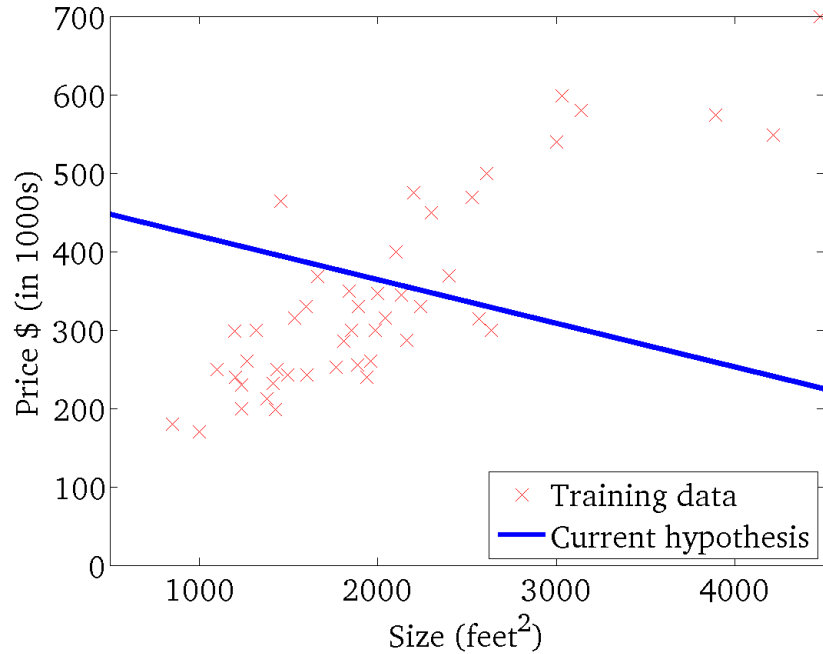
$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



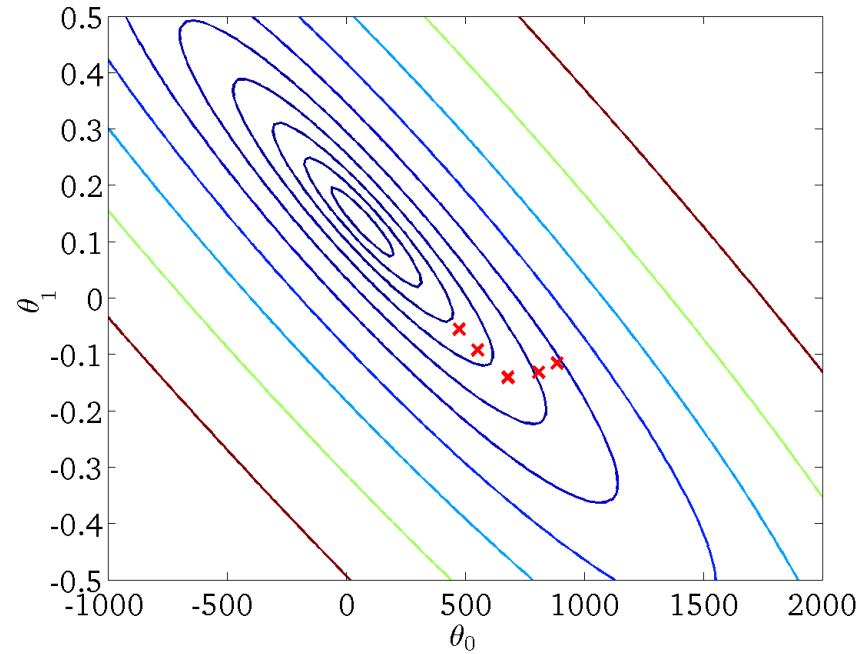
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



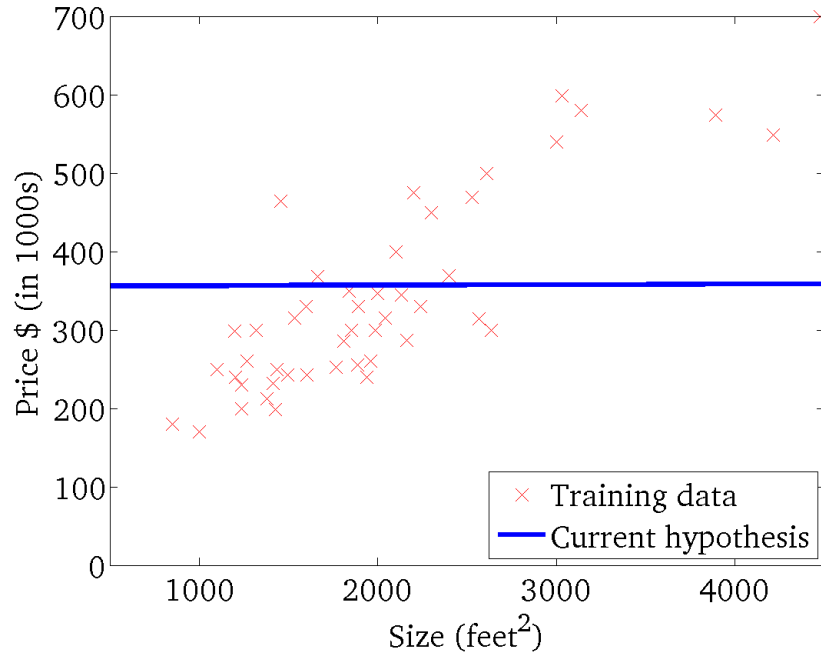
$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



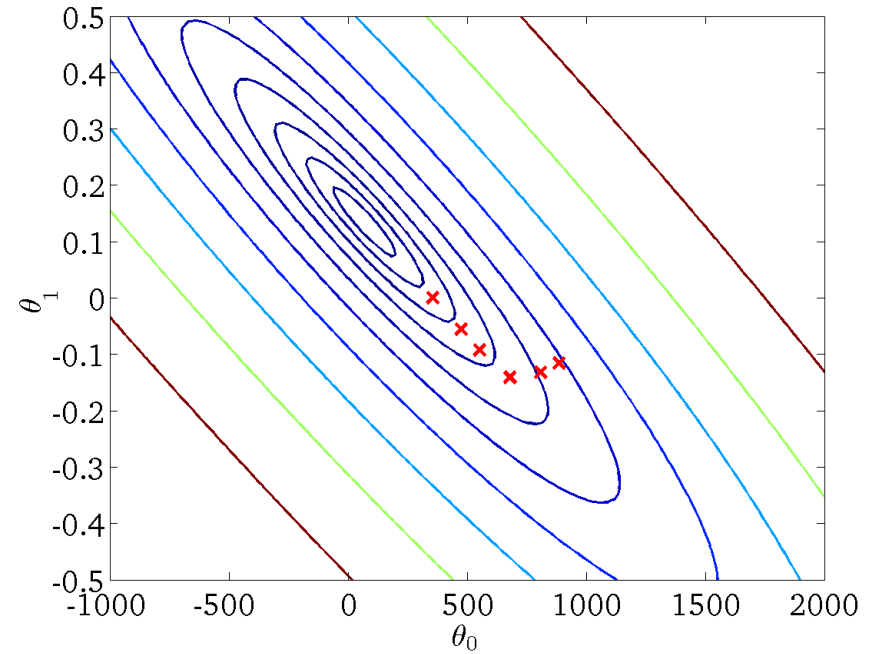
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



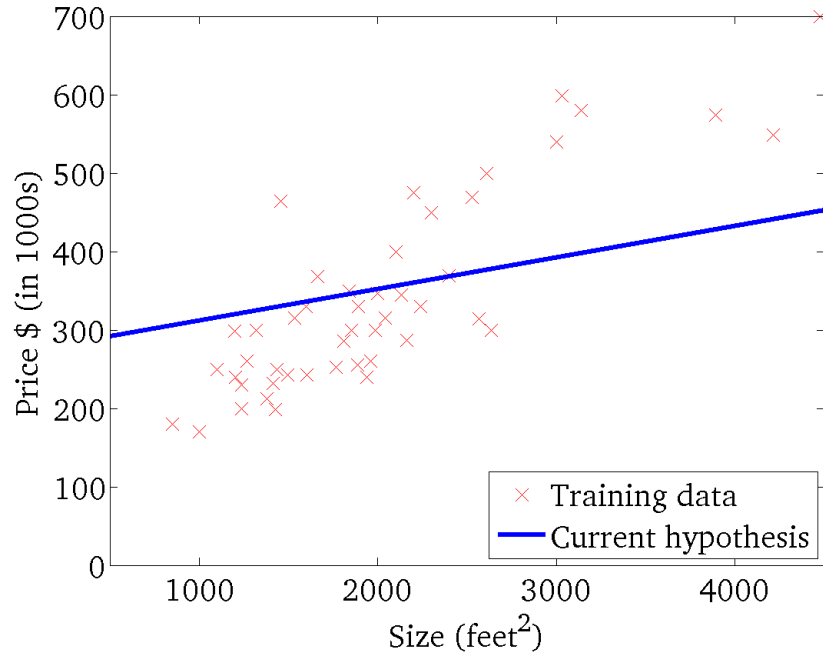
$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



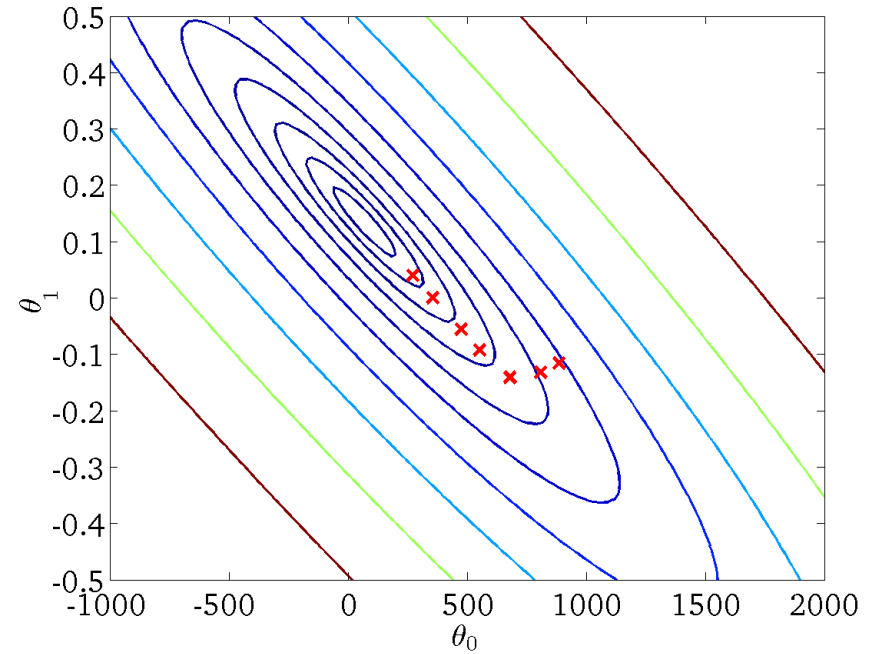
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



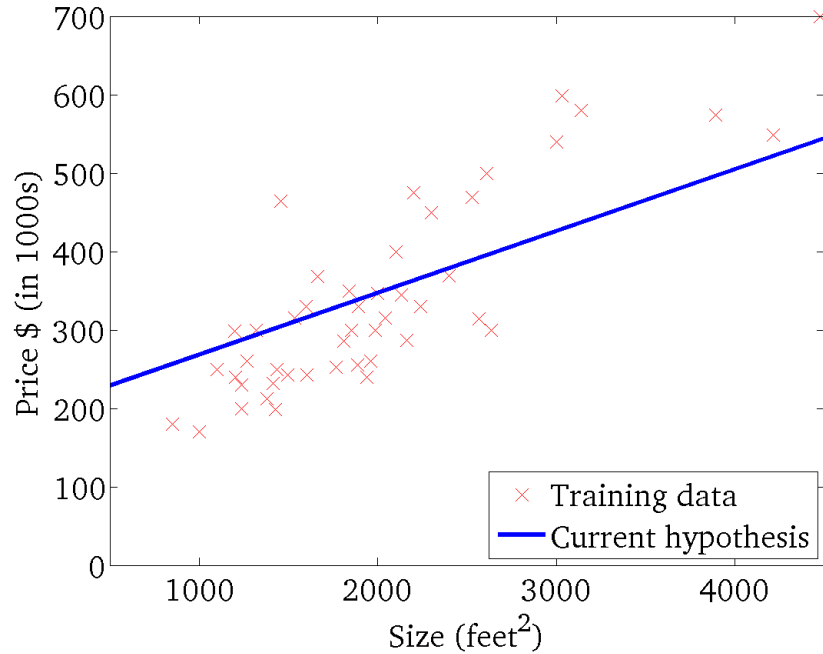
$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



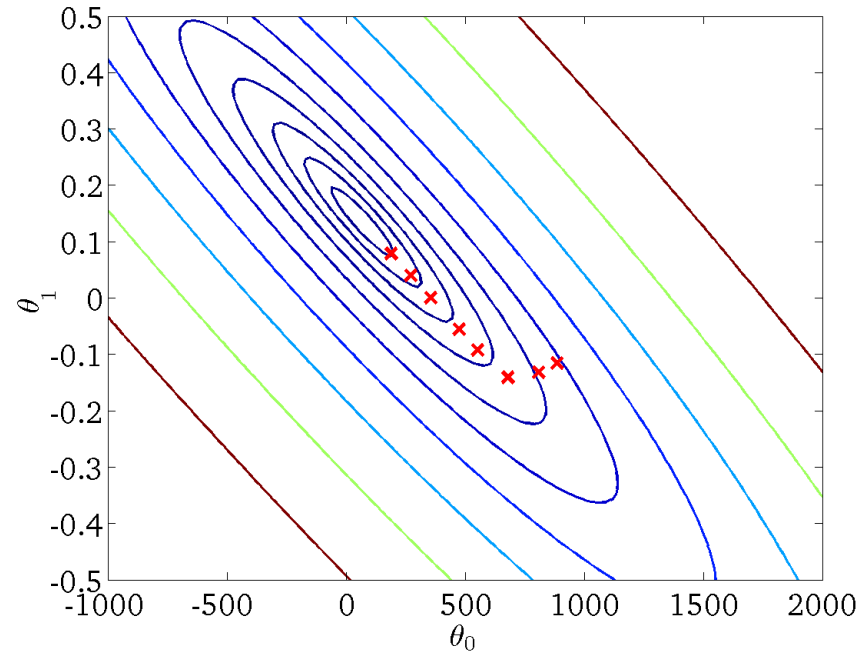
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



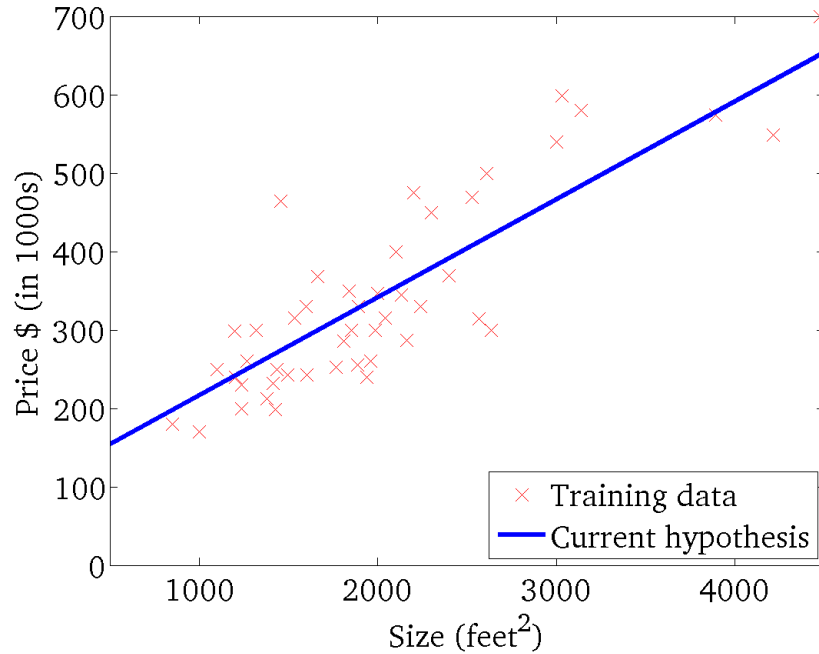
$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



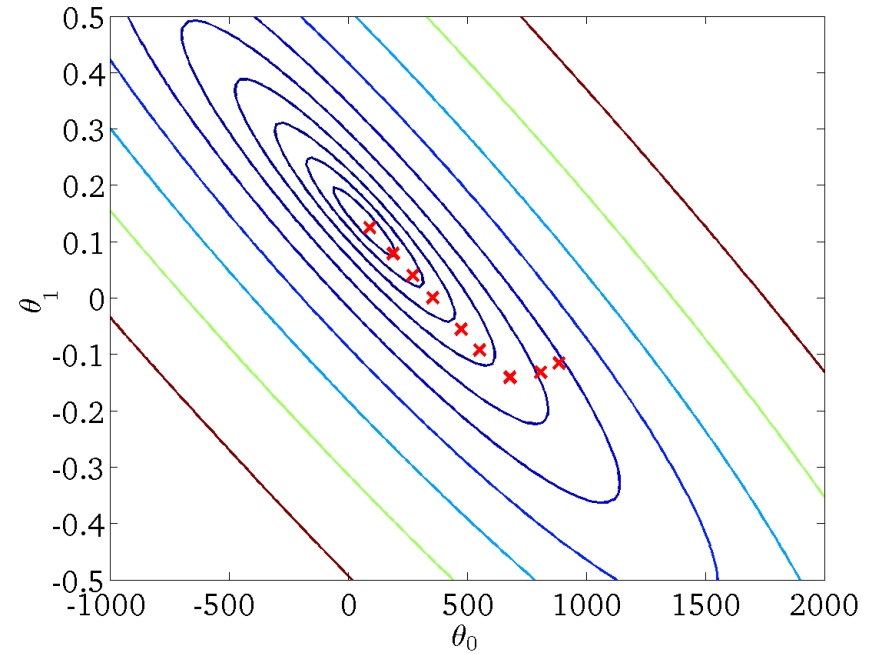
$$h_{\theta}(x)$$

(for fixed θ_0, θ_1 , this is a function of x)



$$J(\theta_0, \theta_1)$$

(function of the parameters θ_0, θ_1)



“Batch” Gradient Descent

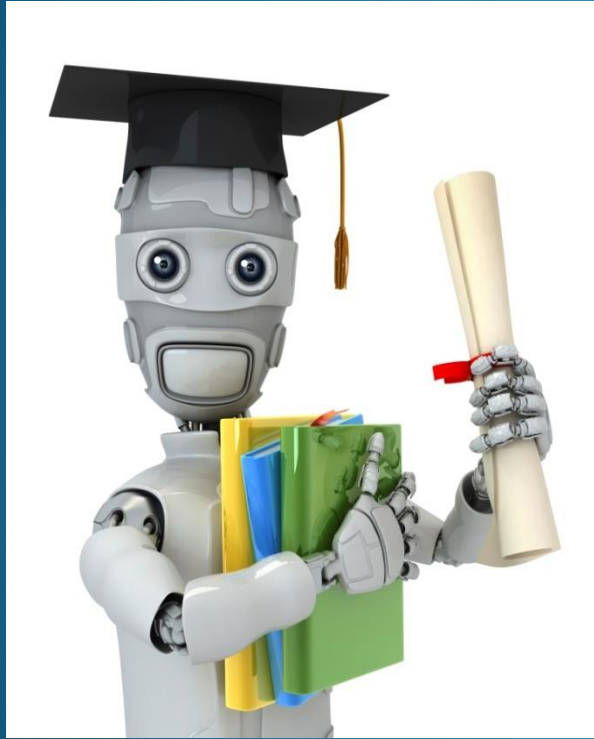
“Batch”: Each step of gradient descent uses all the training examples.

Report

Due date: Next Tuesday

Write a matlab function for the gradient decent algorithm.

Good luck



Thanks